jrg:
nr:

DECEMBER 1983+

++

- ACORN Nieuws -

++



Voor GIRO- en BANKREKENINGNUMMERS zie pag. 2

++

ACORN NIEUWS Uitg.Fed.Acorn Computerclubs Ned/B.
Verschijnt 6-8 x p/jr.

F E D E R A T I E F

voorzitter: A. Millenaar
Thorbeckestraat 38
8161 dp Epe

secretaris: nog even wachten en deze functie wordt vervuld.
tijdelijk correspondentieadres:
FACC
Haringvliet 391
8032 HL Zwolle

penningmeester: G. Visser
Haringvliet 391
8032 HL Zwolle
tel: 038-546561



postgiro: 5244293 bankgiro: 93.32.87.283
Beide tnv penningmeester Acorn Computerclub te Zwolle.

redactie-adres Acorn Nieuws: Henk Reinders,
Leeuwarderstraat 8,
9718 HX Groningen.

Redactie

De aandachtige lezer heeft het reeds gemerkt. Een ander redactie-adres. Het nummer wat u nu leest is het laatste nummer wat van de handen van Ton Otten komt. Een aantal leden van de regio Noord hebben het lofwaardig initiatief genomen om zich in 1984 voor de vervaardiging van uw lijfblad in te zetten. Zie elders in dit nummer.
Wij willen Ton Otten de meeste dank doen toekomen voor de werkzaamheden die hij voor deze (toch redelijk zware) taak verricht heeft. Wij hopen dat hij met even veel plezier de 1984-en-verder exemplaren zal lezen als wij dat met "zijn" spruitelingen hebben gedaan.

Contributie 1984

Onder deze titel is in de vorige Acorn Nieuws ook wat geschreven. Wat daar in stond is nog steeds van kracht. Het enige punt wat daar een beetje overbodig in stond betreft het niet te lang wachten met contribueren. De afgelopen acties van het (semi-) overheidspersoneel hebben wat dat betreft enig roet in het eten gegooid.
Voor de leden die de contibutie voor 1984 nog niet overgemaakt hebben is dan ook de volgende oproep:
Doet u dit zo spoedig mogelijk, zodat een volledig regionaal overzicht niet al te lang op zich laat wachten.
Degenen die om wat voor reden dan ook niet in het bezit zijn van de acceptgirokaart (bv omdat zij later lid zijn geworden) worden verzocht de contributie met een gewone overschrijving te voldoen op een van beide bovenstaande gironummers. Vermeldt u er dan wel even bij dat het gaat om de contributie van 1984 en van welke regio u lid bent?

Archiefdiensten
Voor bandjes, datasheets, EPROM's en alle andere vormen van dienstverlening kunt u terecht bij de regionale archiefdiensten. Mocht in uw regio iets ontbreken, dan kan het regionaal bestuur u verder op weg helpen.
De landelijke archiefdiensten geven regelmatig een overzicht uit van wat er in die archieven aanwezig is. De regionale archiefdiensten kunnen daaruit dan een keus maken van wat zij in het archief willen hebben.

De "clubwinkel"

Geheugenkaart; 16 kByte extra in de Atom	fl. 35,--
Schakelkaart; meerdere EPROM's op Axxx (incl. 74LS133)	fl. 45,--
Programmerkaart; zelf programmeren van EPROM's	fl. 17,--
Herdruk Acorn Nieuws 1982; 97 pagina's wetenswaardigheden	fl. 6,--

Bovenstaande artikelen kunt u bestellen door het desbetreffende bedrag over te maken op een van de gironummers. U kunt natuurlijk ook de bestelling via de regionale penningmeesters laten lopen, zodat u de bestelling veelal tijdens uw volgende bijeenkomst in ontvangst kunt nemen.

Belangstelling gevraagd
Voor het aanvullen van het assortiment in de clubwinkel zouden wij graag willen weten hoe de belangstelling is voor enkele zaken. Zoals daar zijn:
- Big Benny; zie A.N. 1983 nr.4 (met of zonder onderdelenpakket)
- VIA-kaart; voor de schakelaars meerdere VIA's op een print
- telefoonmodem; om via de telefoon Atom's te koppelen
- kleurenkaart; kleuren uit de Atom met evt. geluid en 60/50 Hz.

Degenen onder u die belangstelling hebben voor, aan het ontwikkelen zijn of reeds ontwikkeld hebben van een van deze onderwerpen gelieve dat kenbaar te maken via een briefkaartje aan het redactieadres. Ook wanneer in dit rijtje iets ontbreekt graag even bericht.
In het volgende Acorn Nieuws komt dan een overzichtje te staan van de reacties.

LANDELIJK BANDJESARCHIEF VERPLAATST.

Wegens het overnemen van een bestuurstaak in de regio Noord-holland door N.Stad, vindt er een overdracht plaats van het landelijk bandjesarchief naar een nieuw adres.
De regionale archieven kunnen voortaan nieuwe bandjes bestellen bij:

H. Blijleven
Noordzeestraat 47
1784 BL Den Helder
02230-16734

I N D I T N U M M E R

vervolg: "IN DIT NUMMER"

Bijgewerkt overzicht van de regionale bandjes archieven.

NOORD-NEDERLAND	: J. Homburg Korreweg 185 b 9714 AK Groningen
OOST-NEDERLAND	: F. v/d Biezenbos Burcht 75 7608 JC Almelo
OVERIJSEL/GELDERLAND	: H. v/'t Riet Grevelingen 12 8032 KT Zwolle
UTRECHT E.O.	: H. Seymonsbergen P. de Hooghlaan 37 3741 RL Baarn
NOORD-HOLLAND	: H. Blijleven Noordzeestraat 47 1784 BL Den Helder
DEN HAAG E.O.	: H. v/d Heyden Pr. Annalaan 574 2263 XZ Leidschendam
DELFT E.O.	: P. v/d Berg G. v Oostenstraat 42 2612 RH Delft
ROTTERDAM E.O.	: Mevr. J. Bruins-Sas, Vlaamsestraat 221, 3332 ES Zwijndrecht
ZEELAND	: B. Vundering de Butstraat 47 4561 LT Hulst
ARNHEM E.O.	: G.J. Bouwman R. Kolfschotenlaan 11 6821 JG Arnhem
OOST-BRABANT	: C. Kwakernaak v. Speycklaan 29 5694 CG Breughel
VENLO E.O.	: G. Borghaerts Hatertseweg 3 6581 KD Malden
LIMBURG	: J. Smeets Resedastraat 28 6163 TP Geleen
BELGIE	: R. Leyssens Oudebaan 127 B 3550 Heusden-zolder



.Adres wijzigingen of eventuele fouten in bovenstaande lijst kunnen voortaan aan de heer Blijleven doorgegeven worden.

N. Stad

NIEUWS UIT ROTTERDAM

In verband met de verhuizing van de "oude" coördinator van de Acorn Club Rotterdam, is er in onze regio een nieuwe coördinator. De taken zijn nu als volgt verdeeld:

coördinator	- M. Oosterom	tel: 010-774648
ledenadm./penn.	- R. de Haan	01804-25160
secr./redactie	- J. v. Nieuwenhuizen	01883-15129
drukwerkarchief	- R.D. Urbanik	010-762878
bandjesarchief	- Mw. Bruins	078-123764

Er is besloten voortaan iedere eerste donderdag van de maand bij elkaar te komen, bij Mw. Bruins (Vlaamsestraat 221, Zwijndrecht).

Het DATASHEETS-archief

Hier volgt een overzicht van de in het datasheets-archief aanwezige datasheets. De kopieerkosten bedragen ƒ 0,15 per kopie.

datasheet	blz.	kopie-kosten
6502	10	ƒ 1,50
6520	4	ƒ 0,60
6522	10	ƒ 1,50
6522	24	ƒ 3,60
6530	10	ƒ 1,50
6532	8	ƒ 1,20
2114	4	ƒ 0,60
2716	5	ƒ 0,75
2732	4	ƒ 0,60
8416	5	ƒ 0,75
6847	12	ƒ 1,80
6545	10	ƒ 1,50
8255	21	ƒ 3,15

Bovenop de kopieerkosten komen ook nog de portokosten. Deze bedragen:

aantal kopien	portokosten
0 - 7	ƒ 1,10
7 - 13	ƒ 1,60
13 - 40	ƒ 2,30

De datasheets zijn te bestellen door het totale bedrag over te maken op gironr: 3643518,
t.n.v. G. Akkermans, Wikke 1, Huizen;
onder vermelding van de gewenste kopien.
Dit overzicht is niet helemaal kompleet, dus als je datasheets wil hebben van een ic dat er niet bij staat, laat dit dan even weten!

Gezien in regionale bladen.

Acorn Newsletter nr. 5

Josbox ≠ AXR1: instructie voor instructie vergeleken / Where do we go from here?: Wat te doen nu de Atom niet meer geproduceerd wordt? Antwoord: Zuinig erop zijn en de weinige software-leveranciers te vriend houden! (Uit: "Acorn User", october'83)/ De inktstraaltechniek en de JP101. (Uit: "De mini-microcomputer", october '83)/ Analyse van studietoetsen (Uit: "DATANEWS", 18 october '83)/ Computer-cassettes testen.......Ja, maar!!!: Over goede en slechte cassettes, recorders en hoe ermee om te springen/ Alternatieve Toolbox: het uitbreiden van de Atom met extra statements, op z'n Engels (Uit: "Acorn User" , november '83)/ Speech-synthesizers: een instruktief verhaal over het digitaal opslaan en reproduceren van spraak./ De IEEE 488 bus: Over het koppelen van randapparatuur aan de computer d.m.v. deze universele bus./ Digital logic probe: uitleg bij de datasheet van deze signaaldetector./ Zelf graphics ontwerpen: Over pool- en rechthoekige coördinaten, cirkels, lijnen, etc., voor beginners./ LOGO-demo: een inleiding op deze kinderlijk eenvoudige, gebruiksvriendelijke, maar toch krachtige computertaal./ Eerste ervaringen met een Floppy: met vallen, opstaan en het schakelschema wordt men wijs; een zéér instructief verhaal.

Stacker October '83.

Assembler Bubblesort: een sorteerroutine, in assembler geschreven, met uitleg over de manier waarop het aangepakt wordt/ MIRAKEL verbeterd: een aanpassing van een superkorte disassembler (255 bytes) waardoor deze 3 bytes korter wordt/ Het ELIMIN programma: een oplossingsmethode voor N vergelijkingen met N onbekenden, door de eliminatie-methode (dit is het sluitstuk van een reeks uitstekende wiskundige artikelen voor beginners door J. Woldringh)/ De opvolgers van de 6502-processors: een beschrijving van de nieuwe instructies die beschikbaar zijn op de nieuwe processors 65C02 en R65C02/ Conversietabel voor decimaal- hexadecimaalconversie en omgekeerd (uit: CRC Handboek)/ Berekening van een onbeperkt aantal cijfers in een breuk: een methode om decimale breuken tot vele cijfers achter de komma dóór te rekenen.

Stacker November '83

N! : het berekenen van (een goede benadering van) N! (spreek uit: N-faculteit)/ Telefoon-modem: nu eenvoudig zelf te bouwen door gebruik te maken van een nieuw IC/ Voor hen die er niet uitkomen: Een handige aanvulling op 3D-Maze uit het ATOM MAGIC BOOK, zodat u nu de kluts (dwz. de plattegrond) niet meer hoeft kwijt te raken/ De diskdrive: een uitgebreid CATALOGUS-programma voor diskette, met beschrijving van "hoe en waarom"/ Een disas-lijst van het systeem-programma TRACER/ De opvolgers van de 6502-micropocessors, deel 2: Met een assembleerprogramma voor deze nieuwe processors.

Stacker December '83

Adressering: over het omschakelen van hele geheugenpagina's/ Principeschema van de Schakelkaart/ RHINOCEROS: Een spel waarbij het erom gaat om door een bos met argwanende, bijziende neushoorns heen te komen/ KAT-IN-DE-ZAK: De achtergrond bij een computer-simulatie van bekend beslissingsprobleem: Hoever vertrouw ik mijn concurent?/

Hardware: Over het aan elkaar knopen van verscheidene computers (bij het simulatiespel "KAT-IN-DE-ZAK")/ ASCII-ROUTINE: een routine voor het maken van een tekst-file, voor bewerking door de Tekstverwerker, of voor vastleggen op cassette (voor kopy voor "De Stacker")/ SORT en BPRINT: Twee nieuwe commando's: SORT sorteert een bestand en BPRINT print positieve en negatieve getallen in een aparte kolom.

Cursor nr.6

Het Wachtwoord: Een programma dat vraagt om een geheim wachtwoord in te tikken (dat niet op 't scherm komt!) anders wordt de toegang geweigerd!/ NOISE-VRIJ PLOTTEN: Dit programma wacht het juiste moment af om te gaan plotten, daartoe aangezet door de synchronisatie-impuls van de videoprocessor./ ZERO-PAGE RELOCATOR: Zoekt de door een programma gebruikte ZP-adressen bij elkaar en vervangt ze door andere./ Numeriek toetsenbord: Een subroutine die het toetsenblok 789uijklm transformeert in een standaard numeriek toetsenblok./ A-D-CONVERTER: Een simpele schakeling om de stand van een analoge joystick (= potentio-meter) naar een ATOM-variabele te transformeren./ DOBBELSTEEN: op uw ATOM-scherm, voor als u de echte stenen te onhandig vindt!/ PRINTEN MET MASKER: Een programma dat alleen een bepaalde getalvorm, bijv."ƒ 93,21", accepteert; handig bij het invoeren van geldbedragen!/ Z-Z-STAP INTEGRATOR: Een snel wiskundig integratieprogramma, dat zelf de stapgrootte aanpast aan de gewenste nauwkeurigheid./ SNELLEZEN: Een spelprogramma, waarbij het erom gaat een snel "langslopende" tekst te lezen./ INFOBLOK: Een programma dat titels maakt en herkent bij de supersnelle 3000 en 6000 Baud cassette-lees en -schrijfroutine (zie dit nummer, pagina 15)/ WOORDZOEK: Een soort Mastermind-spel maar nu met woorden van maximaal 10 letters./ JOYSTICK GAMES: De aanpassing van "CENTIPEDE", "PACKMAN" en "PROTECTOR" aan gebruik met joystick./ HET DUPLICEREN VAN PROGRAMMA'S MET DE BBC: De titel zegt 't al./ (Titelloos): Een vergelijking van een tiental micro's op het punt van beschikbare BASIC-commando's.

Het Bronsgroen Eikeltje nr.3

INFOMASTER: Een gedetailleerde beschrijving van het gebruik van dit data-base-programma. Volgende keer volgt de beschrijving van het programma zelf./ ANALOOG-DIGITAAL CONVERTING: Een uitgebreid verhaal, met uitleg voor de niet-ingewijden, van een eenvoudige, goede en goedkope zelfbouw A-D-converter.

Acorntjesbrood nr.3

PRIPRO 1: Eerste van een aantal artikelen over printen van eigen of andermans schermcreaties, zoals spacewuppies, astrobirds of andere fantasiefiguren. Dit eerste deel gaat over print-programma's voor figuren, opgebouwd uit gewone karakters. Met name over de meest handige aanpak van zoiets./ SWEET 16: THE 6502 DREAM MACHINE: Dit is de beschrijving van een interpreter, die een complete set nieuwe (16 bits!) instructies accepteert. Hierdoor verandert uw Atom schijnbaar in een 16-bits machine! De nieuwe instructies zijn eenvoudig mengbaar met "gewone" assembler-mnemonics en BASIC./ SYMBOLISCHE ASSEMBLER voor de ATOM: Deze assembler accepteert echte namen voor labels en variabelen, waardoor assembler-programma's een stuk beter leesbaar worden!/ LINEAIRE NETWERK ANALYSE: Dit programma berekent de frekwentiekarakteristiek van elektronische schakelingen. Een zéér gedegen stuk werk!/

PROM V2.0: Een verbeterde EPROM-programmer; zie dit nummer, pag.26. / STEENKOLENNEDERLANDS+ Over het gebruik van (Engels) jargon in de computerwereld (lees: rekentuig-wereld)/ VERBORGEN 6502 INSTRUCTIES: Over de open plaatsen in de instructietabel van de 6502. Deze instructies doen soms toch wel bruikbare dingen!/ SOFTWARE: FORTH: Eerste van een serie artikelen over de programmeertaal FORTH, maar nu iets meer op de Atom gericht. Dit deel beschrijft aanpassing aan het gebruik van de club 16k RAM-kaart./ SOFTWARE: Geheugen-editor: Een programma om aan de hand van een HEX- of ASCII-dump de geheugeninhoud te veranderen./ SOFTWARE: Zombie: een spelletje, over eten en gegeten worden.

ATOMIX nr 5

DISLIST: Dit programma geeft een disassembler listing in twee kolommen en is dus speciaal geschikt voor A4 papier./ MATRIX INVERSE: een langdurige wiskundige bewerking./ GRAFIC COPY: geeft een hard-copy van het scherm in CLEAR4-mode(eem mooi gestructureerd programma!)/ KIJK MAAR WAT HET DOET: Een pseudo 3-dimensionale tekening met een zeer bekend onderwerp./ KRIJG NOU DE KLEUREN! :Een programma dat, gebruikmakemd van de Delftse kleurenkaart, de Franse Britse en Amerikaanse vlag in kleur op het scherm zet.

ATOMIX nr 6

VIBRATO: Laat de toonhoogte van een opgewekte toon snel of langzaam, veel of weinig trillen./ REGELS SCHOONMAKEN: een zeer snelle manier om een regel op het scherm uit te poetsen./ DISAS in PROGRAMMA'S: Over het gebruik van DISAS-commando's van de Josbox in een programma./ SNOW-TEXT: Laat stippen op uw scherm samensmelten.

Verder bevat dit nummer van ATOMIX een complete opgave van de inhoud van het bandjes-archief, van het drukwerk-archief en van het listing-archief.

GELEZEN: Integer-decimalen. bron:Personal Computer News

Onder deze vage kop geven we hier een tip voor hen die nog steeds geen Floating Point ROM hebben en voor hen die dusdanig veel geheugen nodig (denken te) hebben, dat Floating Point variabelen niet gebruikt kunnen worden.

```
10 @=0;P.$12
20 IN."X"X,"Y"Y
30 A=X/Y;B=X%Y*10
40 C=B/Y;D=B%Y*10
50 E=D/Y;F=D%Y*10
60 G=F/Y
70 P."X/Y="A"."C,E,G
80 END
```

Tip: in plaats van al je beschikbare variabelen op te souperen kun je natuurlijk ook array-elementen gebruiken. Bovendien kun je deze constructie dan in een 'loop' plaatsen.

** OVER REGIOBLADEN ** DE MENING VAN DE (HELE !) REGIO ARNHEM E.O.

WIJ ZIJN EEN VRIJ KLEINE REGIO, MET RELATIEF WEINIG DESKUNDIGHEID.
EEN REGIOBLAD KENNEN WIJ NIET. DE MAANDELIJKSE CLUBAVONDEN STAAN CENTRAAL.
ONDERLING WORDT WEL WAT UITGEWISSELD MET FOTOCOPIEEN. ONTWERPEN EN PROGRAMMA'S, DIE NAAR ONZE MENING VAN BELANG ZIJN VOOR MEER DAN 1 OF 2 LEDEN, WORDEN ALLE NAAR HET LANDELIJKE ACORN NIEUWS GEZONDEN.
VAN DE OPSOMMING VAN 'UIT DE REGIONALE CLUBBLADEN' IN AN 5-83 LEKEN ONS VELE ARTIKELEN VOOR LANDELIJKE PUBLICATIE GESCHIKT. WELLICHT IS HET MOGELIJK, OM HET LANDELIJKE BLAD SIMPELER VAN REDACTIONELE OPZET TE MAKEN: AL HET BINNENKOMENDE 'VOER' ER IN OPNEMEN EN FREQUENTER DOEN VERSCHIJNEN.
MEN HEEFT WELLICHT EEN CONTRIBUTIEVERHOGING OVER VOOR EEN GROTERE HOEVEELHEID INFORMATIE IN BREDERE KRING !!

REDACTIONEEL

Toen ik mij een jaar geleden als landelijk redacteur opwierp, was dat met de uitdrukkelijke bedoeling deze job voor niet langer dan een jaar te doen.
Anders gaat de lol eraf, vind ik.

Welnu, ik prijs mij gelukkig dit schone voornemen realiteit te hebben zien worden door een initiatief vanuit de regio Noord. Een viermanschap gaat het redacteurschap van mij overnemen, waarbij een geschikte taakverdeling de hoeveelheid werk per redacteur aanzienlijk zal kunnen reduceren.

Dit laatste lijkt me te meer een goed idee, omdat ik het eigenlijk wel eens ben met hetgeen de regio Arnhem hierboven schrijft. Het huidige nummer is honderd pagina's dik, en zie in de rubriek "Gezien in regionale bladen" maar eens wat er allemaal <u>nog niet</u> in staat! Daarbij komt nog dat het huidige nummer ruim een maand te laat uitkomt.
Dit laatste heeft overigens andere oorzaken, zoals ziekenhuisopname in December jl.

De nieuwe ploeg staat dus klaar op een goed moment. Met 300% meer mankracht en frisse ideëen over de meest geschikte opzet en aanpak (waaronder automatisering, maar dat zal Henk Reinders u zelf wel vertellen!).

Gaarne wens ik Henk en z'n ploeg alle succes toe en ik hoop dat ze er óók plezier aan zullen beleven, net als ik dat gedaan heb.

uw ex-redacteur: Ton Otten.

Verslag van het Ontmoetingsweekend in Rendeux-Haut.

Op 26/27 november 1983 werd door de Belgische afdeling van de Acorn Computerclub, bij gelegenheid van haar éénjarig bestaan, een ontmoetingsweekend georganiseerd, met het thema:
De ATOM is allang geen ATOM meer; wat nu?

Hiertoe werd de deur van Huize "Vakantiegenot" te Rendeux-Haut wijd open gezet voor Belgische en Nederlandse Atomisten, teneinde elkaar te ontmoeten in:

1. Een wedstrijd: welke regionale afdeling presenteert de beste toepassing van de ATOM?
2. Een debat over het thema van dit ontmoetingsweekend: De ATOM is geen ATOM meer.

Deze activiteiten waren gegarneerd met véél gezellig en informatief samenzijn, veel en lekker eten en afgesloten met een prijsuitreiking met vele prijzen en als verrassende hoofdprijs voor Brabant-Oost: Manneke Pis!
Een korte samenvatting van de wedstrijd en van het debat volgt hieronder.

1. De wedstrijd.
   De volgende 6 presentaties waren aanwezig:

1.1 Twente:
Symbolische Assembler. (o.a. namen voor labels i.p.v. LL∅, LL1)
Voordelen: Betere leesbaarheid van assembler programma's
Binaire getallen kunnen ingegeven worden
Geen schakel- of geheugenkaart nodig
Standaard hardware

1.2 Rotterdam:
a. Hardware: Atom-moederboard in plexi-glas kast ingebouwd tesamen met voeding, EPROM programma, schakelkaart, 2x geheugenkaart, UHF modulator en speaker. Los toetsenbord met aansluitingen voor joysticks. Kast uitklapbaar zodat alles gemakkelijk bereikbaar is.
b. Hardware: Compleet systeem op verrijdbaar audiorack gebouwd.
c. Software: Div. Schakel Operating Systems.

1.3 België:
a. Demonstratie doorlopend laden en verwerken van:
   1. Reclame programma voor ACORN CLUB
   2. Muziekprogramma
   3. Tekening (soort CAD/CAM)
b. Spraakgenerator: Woorden/klanken kunnen zelf "gecomponeerd" worden, zijn echter gebaseerd op Engelse klanken.
c. De ATOM is geen ATOM meer!
   Een geheel andere computer, echter wél gebaseerd op onze trouwe ATOM. Uitgevoerd met:
   2Bestuurbare cassette-recorders d.m.v. DOS-kaart.
   64k RAM in rack & 16k RAM op moederboard.
   RS 232 interface, EPROM programmer, Elektuurbus
d. Demonstratie van de taal LOGO.
   Uitgangspunt: LOGO is een kinderlijk eenvoudige, gebruiksvriendelijke maar toch krachtige

computertaal. Men leert er gestructureerd mee programmeren.

1.4 Overijssel:
Hardware: Een besturingssysteem voor elektromotoren gebaseerd op het tellen van pulsen voor het bepalen van het aantal omwentelingen.

1.5 Oost-Brabant:
a. Software: ∅ Track systeem
Charon, 300, 1200, 3000 Band enz.
b. Adressenbestand met schitterende beeldscherm behandeling.
Laden en "saven" met 6000 band.
c. Het klapstuk van de presentatie:
MDCR = Mini Digital Cassette Recorder.
Een juweeltje van een (inbouw) recordertje voor het laden en saven van programma's op 6000 band op mini-cassettes. De bijbehorende software was uitmuntend en het geheel doet niet onder voor een Floppy-disc-unit. HULDE!!!!

1.6 Delft en Oost-Brabant:
a. Hardware: Een héél goedkope kleurenkaart die nog werkt ook.
b. Software: Microsoft waarmee de Atom geen Atom meer is, doch wordt omgedoopt tot FACTUM.
FACTUM is latijn voor: door mensen gemaakt. FAC staat voor: Federatie Acorn Club.
Deze microsoft zal nog verder worden uitgebreid.

Na rijp beraad kwam een ad hoc benoemde jury van experts tot de volgende rangschikking, waarbij als prijzen (computer)boeken, EPROMS, flessen wijn, cassettebandjes en (heel toepasselijk) een bijna levensgroot beeld van Manneken Pis:

1 Brabant-Oost

2 België

3 Twenthe

4 Overijssel

5 Rotterdam

6 Delft en Oost-Brabant

2. Discussie: "De Atom is geen Atom meer; wat nu?"

De "aftrap" voor deze discussie werd gegeven door ACB-voorzitter Jacques Mijngheer, die de vergadering de vraag voorlegde: Wat is er van de Atom over?
Dhr. Borghaerts ging hierop in door zich af te vragen waaraan je kunt afmeten of iemand of iets (de Acorn Atom) nog dezelfde is als vroeger. Naamsverandering, andere behuizing, andere kleding, het doet er allemaal weinig toe. Waarin schuilt dan het wezenlijke van de Atom? Je kunt eraan sleutelen wat je wil (hardware), maar met software veranderingen moet je voorzichtig zijn, anders verandert het innerlijk, het wezen van de computer. Zo heeft hij zijn apparaat Factum genoemd (Latijn voor:"met handen gemaakt"). Uiterlijk lijkt het op een Acorn Atom, maar het gedraagt zich wezenlijk anders als Factum. Michel van Leuven ging nog een stapje verder met zijn eigen computer, die dermate flexibel is gemaakt dat vrijwel alle voor de

6502- processor geschreven software hierin kan "draaien", dus ook VIC-20programma's. Hij komt tot de conclusie dat de Atom in wezen een stuk gereedschap is. Jacq. Mijngheer constateerde dat Michel kennelijk aan orgaantransplantatie doet: het hart van de een, de longen van een ander etc.
Vervolgens werd meer in detail getracht aan te geven waarin nu precies het wezenlijke van de Atom school: in de computertaal, in de uiterlijke vormgeving (een "ruif" met losse printen, een los toetsenbord), in de geheugenindeling, de Atom-software?
Geconcludeerd werd dat het wezenlijke van de Atom wordt belichaamd in de Atom-ROM, in zijn eigen(wijze) BASIC-dialect.

Vervolgens bracht Jacq. Mijngheer de discussie op de grote groep clubleden, die nog gelukkig is met de oude Atom. Dhr. Borghaerts voegt hieraan toe dat deze grote groep alleen kan profiteren van de verdiensten van de eerder genoemden (de wegbereiders), indien deze laatsten compatibel blijven, d.w.z. overdraagbare software en hardware ideëen lanceren.
Vrijwel iedereen is het erover eens dat de uitwisseling van ideëen en resultaten, het elkaar vooruit helpen met de eigen inzichten en vaardigheden, dat juist dàt hetgene is dat door onze computerclub wordt nagestreefd. Dat niemand zich hierbij hoeft te schamen voor zijn eigen programma's, hoe simpel ook, spreekt vanzelf. Samen met de "cracks" vormen we een RIJKE vereniging, waarin vrijwel alles mogelijk is!

Jan Lernout besluit de discussie aldus:
"Het thema is: "De Atom is geen Atom meer", maar volgens mij is hij het nog altijd.
Wij bestaan nu een jaar als federatie. In het begin werden er vragen gesteld: hoelang zal dit nog duren? We zijn toch helemaal afhankelijk van de Acorn-fabriek?
Maar: wij zijn onafhankelijk geworden. De Atom is van ons geworden. Al bestaan er grote verschillen in ontwikkeling,de vriendschap blijft bestaan. Als anderen veel verder zijn dan ik, kan ik bij hen te rade gaan. Het is een stuk gereedschap. Wij zijn niet te oud om te leren,"

Als redakteur van dit verslag sluit ik me graag aan bij de uitspraak van Mevr. Bruins die mij haar aantekeningen van deze discussie toezond:
"HEEL HARTELIJK DANK aan de Belgische afdeling en ik denk dat ik ook namens de anderen schrijf. Het was in alle opzichten uitstekend verzorgd en geslaagd.
Hopelijk TOT ZIENS!!!!!"

Ton Otten

p.s. Tevens mijn dank aan Ed Schijf, die mij zijn overzicht van de regio-presentaties toezond.

INFOBLOK door: PETER EHRLICH.

Het programma voor supersnel laden en saven van Atomprogramma's op cassette, zoals beschreven in Acorn Nieuws 4/'83 (3000baud) en Acorn Nieuws 5/'83 (6000 Baud), gebruikt voor het identificeren van de geregistreerde datablokken volgnummers, geen namen. Hierdoor is het noodzakelijk bij elk bandje een lijst bij te houden van programma's en bijbehorende bloknummers. Raakt deze lijst na verloop van tijd zoek, of u vergeet een programma toe te voegen, dan zit u in de problemen.

Het volgende programma INFOBLOK schrijft vóór elk te saven programma een blok met de naam van het programma en eventuele verdere toelichting op de band. Dit blok kan gebruikt worden voor het op "op naam" laden van een programma. Ook een CATalogus kan worden gemaakt, die van elk geregistreerd programma de naam en de eventele extra toelichting op het scherm zet.

Het programma INFOBLOK heeft een lengte van 3/4 Kbyte en vormt één geheel met de taperoutine voor 3000/6000 Baud. Begint bijvoorbeeld INFOBLOK op ≠7ØØØ, dan begint de taperoutine op ≠73ØØ en eindigt op ≠75EØ.

Voor het opslaan van de programmanaam en de overige toelichting heeft INFOBLOK een werkruimte nodig van 1/4K; na RUN vraagt het programma om het begin-adres hiervan (V-R?). Vervolgens vraagt het programma: 3 of 6KB ? Antwoord met "6" of "3" (niets mag ook; geeft 3KB). Dan vraagt het programma "L/S/C?" voor LOAD/SAVE/CAT. Ingeval van LOAD of SAVE vraagt het programma verder om een naam (N:?) en het beginadres (B:?). Ingeval van SAVEN gaat het programma door met vragen: eindadres (E:?), startadres (S:?),(deze worden door het programma niet gebruikt), bloknummer (BLOKNR?) en eventuele toelichting.
Vult u geen eindadres in, dan zoekt INFOBLOK dit zelf op.

Vervolgens kunt u een toelichting bij het te SAVEN programma intikken. Een lege string (=RETURN) vormt het einde van de toelichting. De maximale lengte van de toelichting is ongeveer 1/4K; gaat u hierboven dan wordt achteraf de toelichting afgekapt tot 1/4K.

Het programma is natuurlijk veel praktischer in EPROM, maar wordt alvast gepubliceerd tot "leringhe ende vermaeck" en voor eventueel opbouwende kritiek.

```
List
   5in."v-r"p:s=p:n=s+16
   8in."3 of 6kb "$s
   9L=#165:if?s=6:L=#16b
  10in."L/s/c"$s
  20if?s=ch"L"or?s=ch"s":g.a
  25!#a8=p:!#aa=0:!#ac=1
  30Li.((?18+3)*256+L)
  40p.$12"n: "$(p+16)'
  50q=p+Len(p+16)+17
  60p."b: "&!p'"e: "&4!p'"s: "&8!p''
  80dop.$q':q=q+Lenq+1:u.?q=13:g.25
 100ain."n:"$n,"b:"b
 110if?s=ch"L":g.b
 120e=0:in."e:"e
 130ife=0:q=b:doq=q+Lenq+1:u.?q=255:e=q
 140in."s:"s:!p=b:p!4=e:p!8=s
 150in."bLoknr"s:p!12=s:q=p+16
```
REM Voor vervolg zie p.30!

Ton Otten
J.A. de Gravenlaan 17
2381 TA ZOETERWOUDE

Cees van Driel
Walcherseweg 220
4334 NB MIDDELBURG

Middelburg, 11-9-1983

Beste Ton,

Hierbij stuur ik je een eerste overzicht van de zaken, waarmee we in Zeeland bezig zijn; dit voor de rubriek

BRANDPUNTEN

In willekeurige volgorde:

1. Simulatie van processen en geregelde kringen op het gebied van meet- en regeltechniek, met nadruk op het gebruik van de programmatuur in het onderwijs.

2. Ontwikkeling van zeevaartkundige programma's, zowel voor praktisch gebruik als voor didaktische doeleinden.

3. Toepassing van de computer in de communicatie:

   - morse-decodering
   - rtty-decodering
   - ontvangst van weerkaarten en satellietfoto's
   - ontvangst van telexsignalen volgens het TOR-systeem (= Telex over Radio, dat in de scheepvaart wordt gebruikt)
   - de computer als automatisch alarmtoestel voor het registreren van in nood verkerende schepen
   - morsetrainer

4. Toepassing van de computer in de wiskunde

5. Ontwikkeling van een goede kleurenkaart

Zoals ik tijdens ons telefoongesprek heb beloofd, zal ik binnenkort een artikel schrijven over de morsetrainer, die inmiddels is uitgetest op de zeevaartschool in Vlissingen en die mij interessant lijkt voor zendamateurs, die willen leren seinen en opnemen. Tevens zal ik dan nog wat kanttekeningen plaatsen bij zaken als morse- en rtty-decodering.

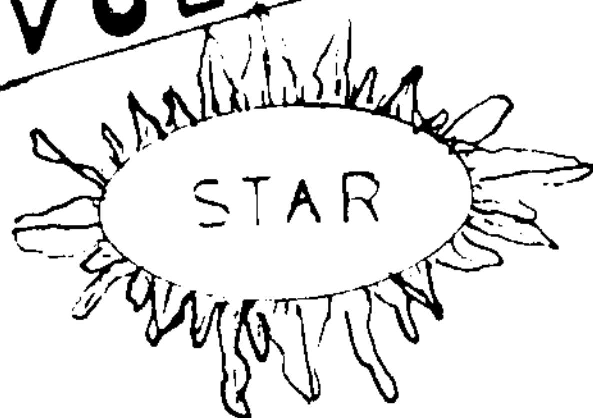
Met vriendelijke groeten,

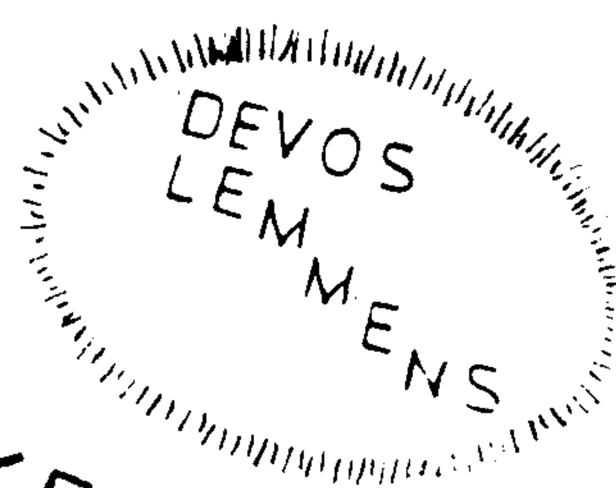
Cees van Driel

```
>L.
   10 REM MJCOS SOURCE CODE 21/08/83
   20 REM (C) M.M.H.M. JANSSEN
   30 REM       BUITENSTEDE 30
   40 REM       3431 XB  NIEUWEGEIN
   50 DIM A1,DD6,SS11,XX0,LL22,II1,FF4,GG20,TT0,PP13
   60 FOR I=0 TO 85; DDI=#8000; N.
   70 P.''"EVEN PROEF DRAAIEN..."$21
   80 I=#9500;O=#1234;GOS.b;GOS.b;P.$6;O=0
   90 FOR I=#9500 TO P-1;O=O+(I-#94FF)*?I;N.
  100 IF O<>#2668D5 P.'"SOURCE CODE IS FOUT"; E.
  110 P.''"assembleer mjcos"''"GEEF BEGINADRES..."; IN.I
  120 IN.'"AANPASSEN VOOR ROM--J/N",$A
  130 IF ?A<>CH"J" O=0
  140 IF ?A =CH"J" IN.'"GEEF BEGINADRES IN ROM",O; O=O-I
  150 P.'"WIL JE EEN LISTING?"''"1 = SCHERM"'"2 = PRINTER"
  160 IN.'"(RET) = GEEN LISTING",$A
  170 P.'"EVEN GEDULD A.U.B."'
  180 P.$21; GOS.b; IF ?A=CH"2" P.$2$6
  190 IF ?A=CH"1" P.$6
  200 GOS.b; P?0=-68; P?1=-34; P?2=-18; P?3=-8
  210 P?4=#20; P?5=#40; P?6=#80; P?7=0; @=3
  220 FOR I=0 TO 7
  230 P.'"          "&P+I,&P?I; N.; @=8;P=P+8; P.'$6$3
  240 P.'"MJCOS STAAT IN HET GEHEUGEN"'"EIND ADRES "
  250 P."VOOR *SAVE:"&P; END
  260b P=I
  270 B=#B002; C=#C0; D=#C4; E=#C7; F=#C8; G=#C9; H=#CB; J=#CC
  280 K=#CE; L=#CF; M=#D1; N=#D3; Q=#D5; R=#D8; S=#D9; T=#DC
  290 U=#F876; V=#B800; W=#F893; X=#EC; Y=#C3; Z=#206
  300[
  310:DD0 LDA Z+1;CMP @(DD3+O)/256;BEQ DD5;STA#FC;LDA Z;STA#FB
  320 LDA @(DD3+O)&#FF;LDX @(DD3+O)/256;:DD1 STA Z;STX Z+1
  330 LDA @#34;BNE DD4;:DD2 LDA#FB;LDX#FC;BNE DD1
  340:DD3 LDY @0;STY B;JSR U;INY;CMP@#30;BEQ DD2;BCC DD6
  350 CMP@#35;BCS DD6;:DD4 TAX;LDA PP13+O-49,X;STA E
  360 LDA PP13+O-45,X;STA F;:DD5 RTS;:DD6 LDX#100,Y;INY;STY N
  370 CPX@#2E;BNE XX0;CMP@#58;BEQ SS0;CMP@#53;BNE LL0;CLC
  380\
  390:SS0 ROR H;JSR TT0+O;LDA#EA;BNE SS1;LDA@6;JSR#FC45
  400:SS1 LDY N;:SS2 BIT H;BPL SS3;JSR U;CMP@#22;BNE SS4
  410:SS3 JSR#F818;:SS4 LDX@L;JSR W;LDX@M;JSR W;BNE SS5
  420 BIT H;BMI SS7+1;LDA#12;STA L+1;LDA#D;STA M;LDA#E;STA M+1
  430 LDA@#B2;LDX@#C2;BNE SS6;:SS5 BIT H;BMI SS8;LDX@J
  440 JSR W;BNE SS7;LDA L;LDX L+1;:SS6 STA J;STX J+1
  450:SS7 LDA@0;STA K;:SS8 JSR U;CMP@#2C;BEQ SS9;DEY;CLC
  460:SS9 ROR H;INY;JSR FF0+O;SEC;ROR H;JSR U;CMP@#D;BNE SS2
  470:SS10 LDA@0;STA V+11;:SS11 RTS;:XX0 JMP(#FB)
  480\
  490:LL0 CMP@#43;BEQ LL2;CMP@#4C;BEQ LL2;CMP@#56;BEQ LL2
  500 CMP@#52;BNE XX0;LDA@0;:LL2 LSR A;ROR A
  510 ROR A;STA R;JSR#F818;LDX@Q;JSR W;LDA#EA;BNE LL4;LDA@4
  520:LL3 JSR#FC45;:LL4 SEC;:LL5 ROR S
```

```
530:LL6 JSR GGØ+O;BCC SS11;LDX@Ø;STX T;:LL7 JSR GG1+O
540 STA#ED,X;CPX@13;BEQ LL8;INX;:LL8 CMP@#D;BNE LL7;LDX@8
550:LL9 JSR GG1+O;STA G+1,X;DEX;BNE LL9;JSR GG1+O;BIT R
560 BVS LL11;LDA T;BNE LL13;BIT S;BPL LL16;LDY@#FF
570:LL1Ø INY;LDA(G),Y;CMP@#D;BEQ LL14;CMP#ED,Y;BEQ LL1Ø
580:LL11 LDY@Ø;JSR#F992;LDX@L;JSR#F7EE;LDX@J;JSR#F7F1;LDA K
590 JSR#F8Ø2;JSR#FFED;:LL12 CLC;BCC LL6;:LL13 JMP#F9FD
600:LL14 BIT H;BVC LL15;LDA@5;BNE LL3;:LL15 LDA Q+2;BNE LL17
610:LL16 LDA L;STA Q;LDA L+1;STA Q+1;:LL17 BIT S;BPL LL18;CLC
620 ROR S;LDA J+1;CMP@#C2;BNE LL18
630 LDA L+1;STA#12;:LL18 JSR IIØ+O;:LL19 INY;JSR GG1+O
640 BIT R;BMI LL2Ø;STA(Q),Y;BPL LL21;:LL2Ø CMP(Q),Y;BNE LL13
650:LL21 CPY#D4;BNE LL19;JSR GG1+O;LDA T;BNE LL13;INC L+1
660 INC Q+1;TXA;BNE LL18;LDA H;BMI LL12;LDA J+1;CMP@#C2
670 BNE LL22;JMP#CD9B;:LL22 LDA R;BNE II1;JMP(J)
680\
690:IIØ LDY@Ø;STY T;DEY;CLC;LDA M;SBC L;STA#D4;LDA M+1
700 SBC L+1;TAX;BEQ II1;STY#D4;:II1 RTS
710\
720:FFØ STY N;JSR#FB81;LDY@Ø;STY T;:FF1 LDA(G),Y;JSRPPØ+O;INY
730 CMP@#D;BNE FF1;LDX@8;:FF2 LDA G+1,X;JSR PPØ+O;DEX;BNE FF2
740 LDA T;JSR PPØ+O;:FF3 JSR#FE71;JSR IIØ+O;:FF4 INY;LDA(L),Y
750 JSR PPØ+O;CPY#D4;BNE FF4;LDA T;JSR PPØ+O;INC L+1;TXA
760 BNE FF3;INC K;LDY N;RTS
770\
780:GGØ STY Y;LDY@#8Ø;BNE GG2
790:GG1 STY Y;LDY@#FF;:GG2 STX X;LDX@Ø;SEC;LDA@#2Ø;AND B;STAD
800 BEQ GG11;BNE GG13;:GG3 CPX E;BCS GG4;LDY@#8Ø;BIT#BØØ1
810 BVC GG2Ø;:GG4 CPY@#FF;SEC;BNE GG8;BEQ GG19;:GG5 CPX E
820 BCC GG6;LDY@#FF;:GG6 BCC GG7;:GG7 NOP;:GG8 LDX@#FC;EOR D
830 STA D;:GG9 BNE GG1Ø;:GG1Ø BNE GG13;:GG11 LDA@#2Ø
840:GG12 DEX;BIT B;BEQ GG12;BCS GG14;CPX E;BCC GG15
850:GG13 DEX;BIT B;BNE GG13;BCS GG14;CPX E;BCC GG15;BCS GG12
860:GG14 INY;BMI GG3;BEQ GG5;DEY;BCS GG16;:GG15 EOR D;STA D
870:GG16 ROR C;LDA C;ROL A;ROL A;AND@1;EOR T;LSR A;BCC GG17
880 EOR@#AE;:GG17 BCC GG18;:GG18 STA T;NOP;LDX@#F9;INY;CPY@8
890 LDA D;BCC GG9;:GG19 LDX X;LDY Y;LDA C;STA#8ØØØ;RTS
900:GG2Ø CLC;RTS
910\
920:TTØ LDA@#CØ;STA V+11;LDA@Ø;STA V+5
930\
940:PPØ PHA;STA C;STX X;STY Y;LDX@#FF;CLC;LDA V+4
950:PP1 LDA@#4Ø;:PP2 BIT V+13;BEQ PP2;STA V+13;BCC PP4
960 CPX@9;BCC PP2;LDX X;LDY Y;PLA;RTS
970\------------------------------
980:PP4 INX;BEQ PP5;SEC;ROR C;:PP5 PHP;LDA@52;BCS PP6;ASL A
990:PP6 STA V+6;LDY F;BEQ PP8;STY V+7
1000:PP7 ASL V+6;ROL V+7;BCC PP7;:PP8 CPX@8;BCS PP1Ø;LDA C
1010 AND@1;EOR T;LSR A;BCC PP9;EOR@#AE
1020:PP9 STA T;:PP1Ø PLP;JMP PP1+O
1030:PP13\ 8 BYTES OF DATA
1040]
1050 RETURN
```

OVER VOEDINGEN GESPROKEN !





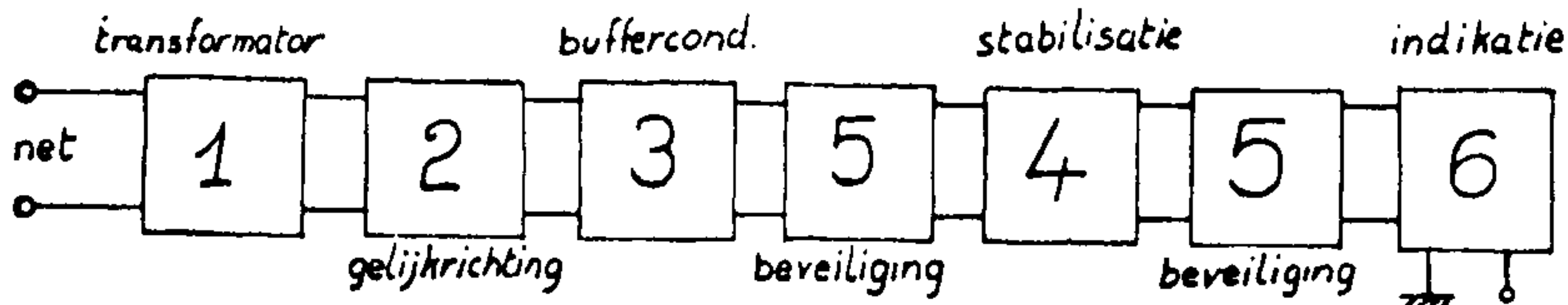
Een belangrijke zaak voor een computer is de voeding, hoe eenvoudig de opbouw ervan, ook kan zijn. Bij een slecht werkende voeding kunnen er heel wat storingen optreden :

- een in spanning schommelde voeding kan o.a. het uitwissen van geheugen blokken tot gevolg hebben ;
- brom (rimpel) geeft een golvend beeld, dansende letters ;
- netstoringen, die over de voeding binnenkomen kunnen IC beschadigen of een foutieve "klokking" veroorzaken.

(zie hieromtrent ook artikels in A.C.-Nieuws)

Hierna een zeer korte toelichting omtrent de verschillende bestanddelen van een voeding aan de hand van het klassieke blokschema.

BLOKSCHEMA VAN EEN VOEDING



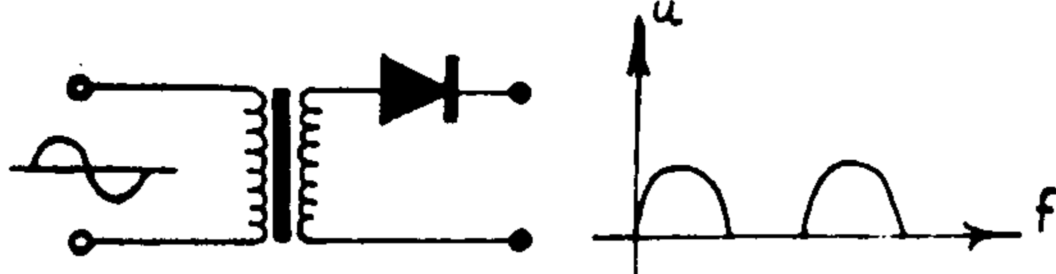
1. Transformator : dient om de netspanning om te zetten naar een lagere waarde en galvanisch te scheiden.

   nadeel : - voor 50 Hz groot en zwaar
   - kan een magnetische veld opwekken dat hinderlijk kan zijn voor de randapparatuur.

   soorten : - de klassieke E-kern ;
   - de ringkerntransformator : waarbij de eerder genoemde nadelen wat verminderd worden.

   opmerking : de opgegeven spanning bij een transfo is de EFFECTIEVE waarde.

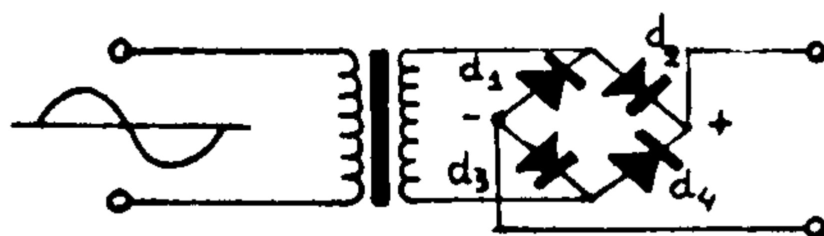
2. Gelijkrichting : zet de wisselspanning om in een pulserende gelijkspanning

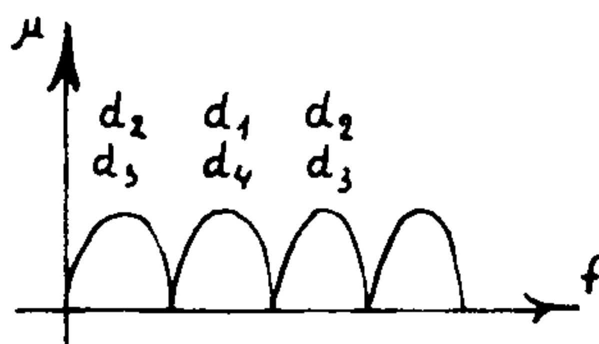


eigenschappen :-gemiddelde diodestroom 100 % (alle I gaat door de diode) ;
-rimpelfrequentie is 50 Hz.

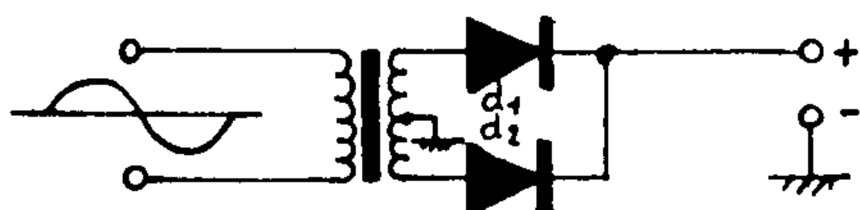
dubbelfazige

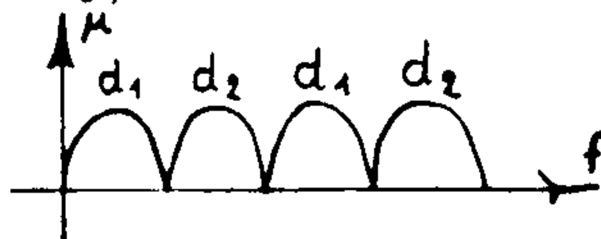
1. met bruggelijkrichter





2. met center tap (midden aftakking)

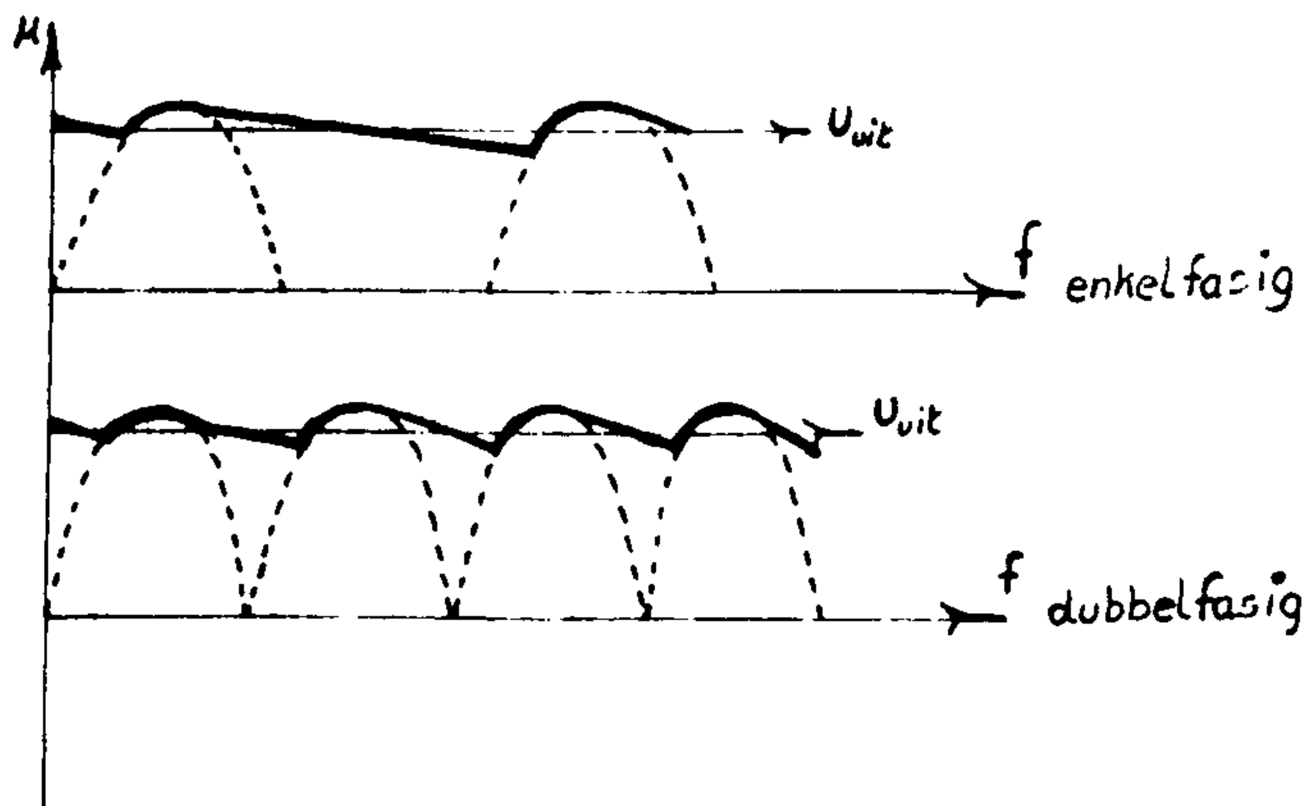




eigenschappen : -gemiddelde diode-stroom 50% van de totale stroom
-rimpelfrequentie 100 Hz ;
- voordeel : centertap : er zijn slechts 2 diodes nodig
- nadeel : transfo is wel duurder

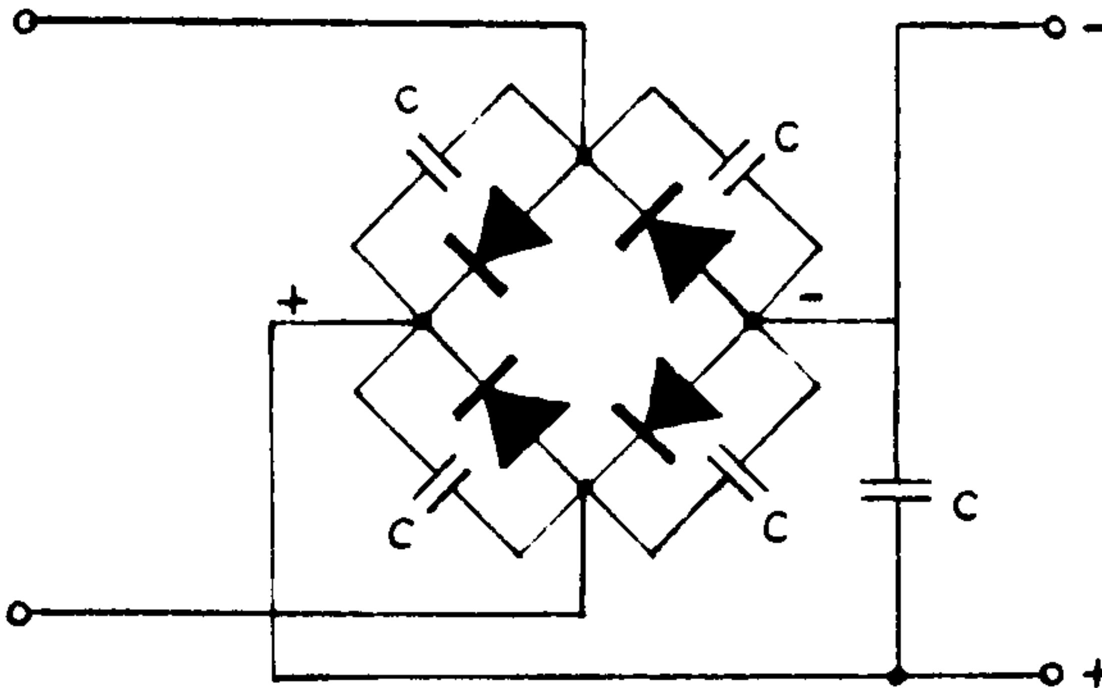
3. Afvlakking ;

- doet dienst als buffer
- bij enkelfasige gelijkrichting met condensator heeft men ongeveer de effectieve spanning ;
- bij dubbelfasige gelijkrichting heeft men ongeveer de maximale "piek"waarde $U_p = U. \sqrt{2}$



Opmerking :

Waarom zijn er bij sommige bruggen condensatoren Parallel over de diodes ? d.i. voor TRANSIENT SUPPRESSION (doorgang - onderdrukking) : pieken veroorzaakt bij ingeschakelde motoren, dimmers, ballasten van T.L.-lampen , de inverse herstelcurve bij gelijkrichterdiodes, inschakelen van de voeding. Deze condensatoren zijn voor de pieksp. (=hoge frequenties) een kleine weerstand, waardoor deze pieken over de condensator hun weg vinden en niet door de diodes. De diodes worden op deze manier beschermd om de maximum inverse spanning niet te overschrijden.



4. Stabilisatie

Een gelijkgerichte spanning is niet constant te houden om volgende redenen :

- schommeling in de netspanning ;
- veranderende belasting die spanningsschommelingen veroorzaakt in de gelijkrichter ; (dit kan ook een verandering geven van de rimpelspanning)
- temperatuursinvloed die ook invloed heeft op het gedrag van de componenten.

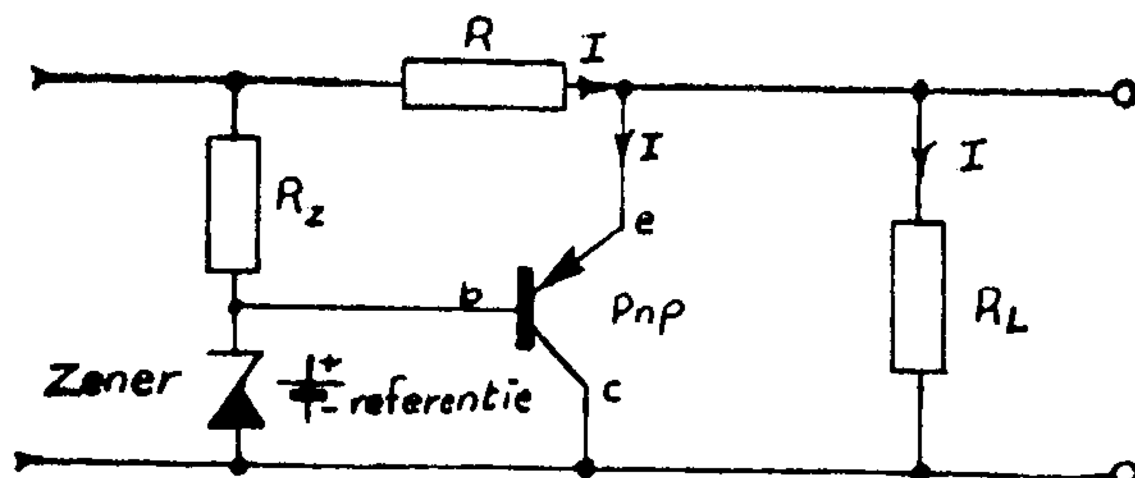
DAAROM STABILISATIE !!!

Naargelang : 1. Principeopstelling : serie of parallel ;
2. werkingsprincipe van de stabilisatie :
- een lineaire regeling van het schakelend element : de geleiding verloopt lineair van niet-geleiding tot verzadiging ;
- geschakeld : geleidend tot gesperd van het element. (zal in deel twee besproken worden)

lineair	geschakeld
seriestabilisatie	serie
parallelstabilisatie	parallel
(engelse benaming)- lineair voltage regulator	switching regulator

PARALLELSTABILISATIE

Werkingsprincipe :



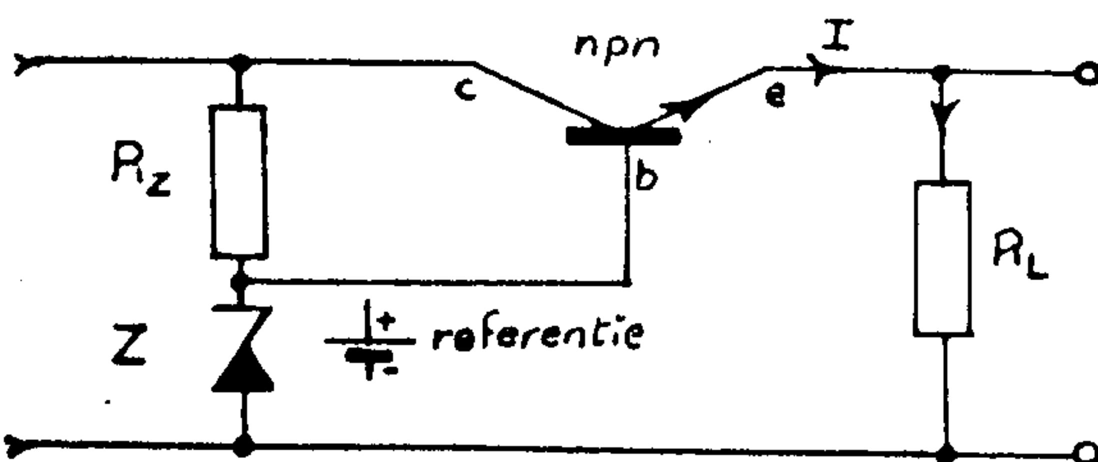
Veronderstel dat de stroom door de belasting (R_L) vermindert. Bijgevolg zal er over (R) minder spanningsval zijn. (U) zal over (R_L) vergroten maar de basis van de PNP-transistor (T) staat op een vast potentiaal. In de emitter sp. hoger dan zal de transistor meer geleiden. Hoe groter (I) door (T) dan (I) groter door (R). De spanning over de (R_L) daalt dus en de (U) over (R_L) blijft gestabiliseerd.

Enkel toepasbaar voor kleine belastingen.

SERIESTABILISATIE

Dit is de meest gebruikte stabilisatie van de twee.

Werkingsprincipe :



De emittervolger is hierin herkenbaar. Verkleint de belasting dan zal de uitgangsspanning verminderen. De NPN-transistor wiens basis op een vast potentiaal staat, krijgt een negatiever emitterspanning. Dit betekent meer geleiden → (I) → zodat de spanning stijgt tot wanneer zich de spanning herstelt op het ingesteld niveau.

Nadelen :
- rendement (η) = 40 %à 50 %
- indien er een uitgangsspanning, nodig is die niet in de handel te verkrijgen is , moet men een transfo kiezen met een hogere uitgangsspanning. Hierdoor ontstaat er wel een groot vermogenverlies.

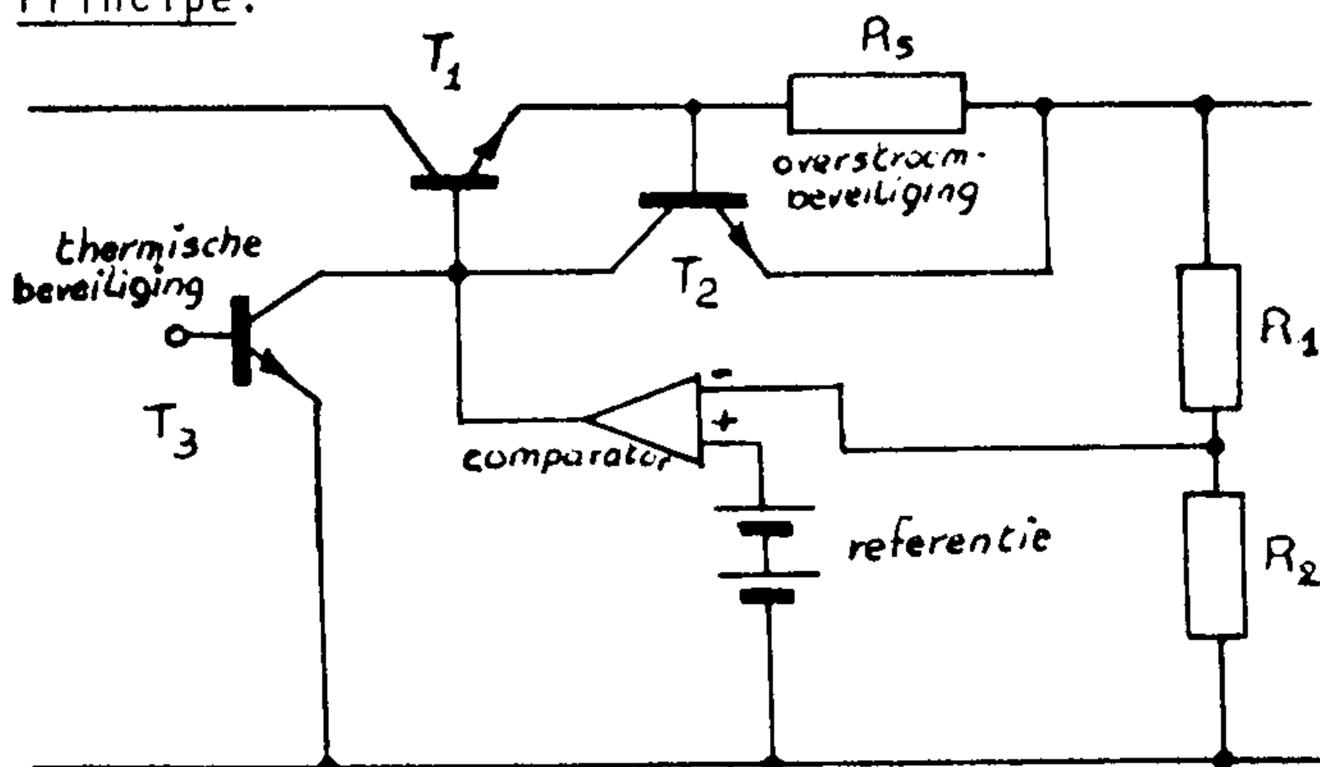
1. transfo 8V, geeft gelijkgericht + 12V
uit $U_{in}-U_{uit}$ → 12-5 = 7V
P = U x I → 5A → 7x5 =35Watt te dissiperen door de transistor

2. transfo 12V, dit geeft gelijkgericht ± 18V
uit U_{in} - U_{uit} → 18-5= 13V
P = U x I → neem 5A → 13x5=65Watt te dissiperen door de transistor

Hiervan wordt er slechts 5Vx5A = 25 Watt te gebruikt en de rest moet gedissipeerd worden (omgezet in warmte). Is nu de voeding opgesteld in de computerkast, of bij de uitbreidingsprinten dan geeft dat onnodige verhitting van de componenten tot gevolg.

Opmerking : het verschil van $U_{in}-U_{uit}$ moet zeker 2V bedragen voor de goede werking van de stabilisatie (transistor of regulator).

De handelsvorm van een serie stabilisator is de
VOLTAGE REGULATOR
Verkrijgbaar in 5A en meer.
Principe.



Werking :

T_1 is de serie-regulator.

T_2 de overstroombeveiliging over R_S krijgt men een spanningsval in functie van de uitgangsstroom. Bij een bepaalde gekozen waarde ($U=R_S x I$) U_{R_S} = UBE zal de transistor T_2 gaan geleiden hetgeen de transistor T_1 beïnvloedt en de uitgangsstroom gaat beperken.
De comparator vergelijkt de referentiespanning met de uitgangsspanning, die bij een afwijking de serietransistor zal bijsturen. (meer of minder doet geleiden).

Meestal is er ook een temperatuur beveiliging. Wordt de chiptemperatuur te hoog dan geleid T_3 hetgeen de basis aan de massa legt waardoor de uitgangsspanning 0V wordt.

Veel gebruikte termen bij de voltage regulator. :

foldback (terugloopkarakteristiek)
foldbackcurrent : is de stroom waarbij de intregrated circuit (I.C.) terugvalt bij kortsluiten van de uitgang. Indien dit niet zou gebeuren zou

de temperatuur (dissipatie) serietransistor, te hoog oplopen en beschadigen.

dropout_voltage (uitvalspanning)

Het verschil van $U_{uit} \rightarrow U_{in}$, beneden de regulator bij een verdere daling van U_{in} de uitgang niet meer stabiliseert. Deze is afhankelijk van de junctie temperatuur (een junctie bij halfgeleiders is het gebied dat zich vormt tussen de twee verschillende materialen) en de belastingsstroom.

quiescent_current (gevraagde stroom)

dit is de stroom gevraagd door de IC welke niet aan de belasting geleverd wordt.

ripple_rejection (rimpelonderdrukking)

de verhouding van $\frac{U_{in}\ rimpel}{U_{uit}\ rimpel}$ uitgedrukt in dB.

line_regulation

de verandering in output (uitgang) spanning voor een verandering in input (ingang)spanning.

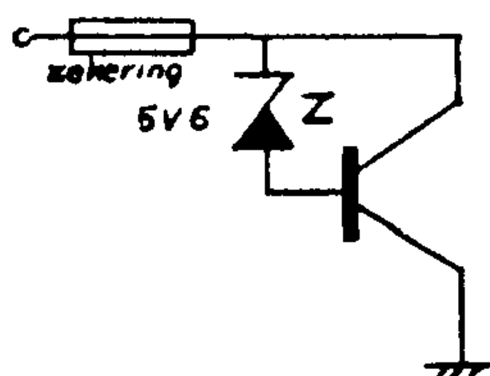
load_regulation

de verandering in uitgangsspanning, voor een verandering van de uitgangsstroom.

5. Beveiliging :

Soorten

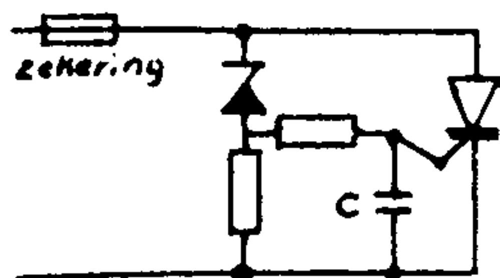
1. met transistor :



werking : neem aan dat de in- en uitgang van de stabilisator inwendig tegen elkaar komen, of de serietransistor in een gewone voeding komt in kortsluiting De spanning zal gelijk zijn aan de ongestabiliseerde spanning ; indien deze groter wordt dan 5V6 (zenerspanning) + 0,7 V (Ube-junctie) = 6,4V, de collectorstroom heeft de grootorde van de maximum leverbare stroom van de transfo. Is er geen zekering opgenomen dan gaat dit door tot de transistor rooksignalen geeft.

Conclusie : de pieken worden begrensd op 6,4V en de transistor komt dan in geleiding, die ze als het ware kortsluit.

2. met thyristor :



werking : krijgt men een overspanning door veronderstelde reden van bij de transistor, dan zal de thyrister triggeren bij een spanning van de zenersp. + de triggersp. van de gate (V_g). De thyristor is nu een kortsluiting welke de zekering die opgenomen is in de kring doorbrandt. Zo is de schadelijke spanning onmiddellijk verwijderd van de computer.

-nadeel : bij piekspanning moet er opgelet worden dat de thyristor niet triggert, want anders zou telkens bij opschakelen de zekering doorbranden. Daarom de condensator aan de gate die een kleine tijdsvertraging tewerkstelt.

* variant met thyristor

-nadeel : te traag, relais heeft betrekkelijk veel vertraging voor het afschakelen van de voeding. Dit kan reeds fataal geworden zijn voor de I.C.'s.

opmerking : om bij deze schakeling toch iets te doen tegen de schadelijke pieken, kan men gebruik maken van een zo genaamde Ge-MOV.
Ge-MOV : is een soort zenerdiode die stromen tot 20 A voor korte perioden kan dissiperen.

3. met IC. van Motorola type : MC 3425

beste keus en tevens indikatie van juiste uitgangssp.

gebruikte termen

crowbar (koevoet, breekijzer)

met geweld openbreken van de voedingslijn d.m.v. een thyristor.

electronic shutdown (elektronisch tijdelijk afschakelen)

mogelijkheden

- met relais, onderbreken voor of achter de regulator (transistor) ;
- met zekering (=crowbar) ;
- referentie spanning kortsluiten.

6. Indikatie.

Het is steeds interressant om een indikatie te hebben van de uitgangsspanning. Dit laat toe te weten :

-of er in geval van meerdere voedingen, een spanning ontbreekt.
-of de 5V niet te veel daalt. Anders ontstaan er fouten in de logische niveaus. (door te dunne en te lange koperbaantjes is er reeds spanningsverlies. Komt nu een spanningsdaling, dan is het verschil van de logische '0' en '1' te klein).

Besluit : gezien de gunstige prijs van de LM 3914 is dit de beste keus, zoals beschreven in Elektuur van juli/augustus 1981

- *Rudy GRIJP - A.C.B.*

PROM V2.0

PROM is een machinetaal besturingsprogramma voor de EPROM-PROGRAMMER uit ACORN NIEUWS 7-82 en 1-83.
Ten opzichte van het originele BASIC programma uit PRACTICAL COMPUTING (ACORN NIEUWS 7-82) heeft het de volgende voordelen:

- Het is sneller (4K lezen (1 sec.) behalve bij programmeren, daar wordt de tijd bepaald door de 50 msec/byte die de EPROM nodig heeft voor het programmeren.
- Het is kompakter (588 bytes) zodat het geschikt is om als commando in een prom te worden opgeborgen.
  (bijv. op #EXXX of #AXXX van de schakelkaart.)

Het programma is in source-code weergegeven met redelijk wat toelichting, zodat de "sleutelaars" naar eigen inzicht veranderingen kunnen aanbrengen.
Na RUN vraagt het programma om een assembleeradres vanaf waar de object-code kan worden weggeschreven.
Dit is tevens het opstartadres van het programma.

Gebruikte zero-page adressen (hexadecimaal):

91,92	PROM adres (loopt van #0000 tot #1000)
93,94	RAM adres (loopt van #8400 tot #9400)
95	ingelezen byte
96,97	checksum

COMMANDO'S:

- B - Blank check
- R - Read prom
- S - Checksum
- V - Verify
- W - Write prom
- Q - Quit

MELDINGEN:

INSERT ROM	: rom inzetten
SET read	: schakelaar op read zetten
?	: commando intypen
ERROR	: onbekend commando
BLANK CHECK OK	: PROM is leeg
BLANK CHECK ERROR ON #0000	: eerste adres ()#FF
VERIFY OK	: PROM = RAM
VERIFY ERROR ON #0000	: eerste adress waar PROM()RAM
SET program	: schakelaar op program zetten
PROGRAMMING	: bezig (3 min. 26 sec.)
READY	: klaar met programmeren/lezen
SUM=#0A36	: berekende checksum

Opmerkingen:

- Voor het programmeren van 2K PROMS moet in regel 610 LDA@#10 door LDA@#08 worden vervangen.
- Om vanaf een ander RAM deel te kunnen werken moet in regel 540 LDA@#84 door LDA@#XX worden vervangen, waarbij XX het hoogste byte van de nieuwe RAM lokatie is.

Maarten van Alphen

```
 10 REM PROM V2.0
 20 REM AUTEUR:
 30 REM M.P. VAN ALPHEN
 40 REM WIERDEN (OV)
 50 P.$12
 60 IN."ASSEMBLEREN OP"D
 70 DIMVV35;F.N=0TO35;VVN=D;N.
 80 F.I=1TO2;P.$21;P=D
 90[;\*** PROGRAMMA INITIALISATIE ***
100 LDA@#55;STA#208
110 LDA@0;STA#B80E;LDA@#CC;STA#B80C
120:VV7 JSR#F7D1;]
130 ?P=#0C;$(P+1)="PROM PROCESSOR";!(P+15)=#EA0A0A0D;P=P+19;[
140:VV10 JSRVV0;JSRVV1;JSRVV34;JSR#CD54;JSR#F7D1;]
150 $P="INSERT ROM";P=P+10;[;NOP
160 JSR#FE94;JSR#CD54
170\*** COMMANDO INLEZEN ***
180:VV11 JSRVV0;JSRVV1;LDA@#3F;JSR#CD0F
190 LDA#100;CMP@#51;BNEVV33
200 LDA@0;STA#B80C;LDA@#52;STA#208;JMP#C2CF
210:VV33 CMP@#42;BEQVV12;CMP@#52;BEQVV18;CMP@#53;BEQVV26
220 CMP@#56;BEQVV21;CMP@#57;BEQVV30
230 JSRVV6;JSR#CD54;JMPVV11
240:VV30 JMPVV28
250\*** BLANK CHECK ***
260:VV12 JSRVV0
270:VV13 JSRVV1;JSRVV5;LDA@#FF;CMP#95;BNEVV14
280 JSRVV2;BEQVV13;JSRVV15;JSRVV16;JMPVV11
290:VV14 JSRVV15;JSRVV6;JSRVV17;JMPVV11
300\*** READ (P)ROM ***
310:VV18 JSRVV0
320:VV19 JSRVV1;JSRVV5;LDY@0;LDA#95;STA(#93),Y
330 JSRVV2;BEQVV19;JSRVV20;JMPVV11
340\*** CHECKSUM BEREKENEN ***
350:VV26 JSRVV0
360:VV27 JSRVV1;JSRVV5
370 CLC;LDA#95;ADC#96;STA#96;LDA@0;ADC#97;STA#97
380 JSRVV2;BEQVV27;JSRVV25;JMPVV11
390\*** VERIFY (P)ROM ***
400:VV21 JSRVV0
410:VV22 JSRVV1;JSRVV5;LDA#95
420 LDY@0;CMP(#93),Y;BNEVV23;JSRVV2;BEQVV22
430 JSRVV24;JSRVV16;JMPVV11
440:VV23 JSRVV24;JSRVV6;JSRVV17;JMPVV11
450\*** WRITE PROM ***
460:VV28 JSRVV31;JSR#FE94;JSR#CD54;JSRVV35;JSRVV0;JSR#FE66
470:VV29 JSRVV1;LDY@0;LDA(#93),Y;STA#B800;LDA@#EC;STA#B80C
480 JSR#FE66;JSR#FE66;JSR#FE66
490 LDA@#CC;STA#B80C;JSRVV2;BEQVV29;JSR#CD54;JSRVV20
500 JSRVV34;JSR#FE94;JSR#CD54;JMPVV11
510\*** INITIALIZE ADRESS ***
520:VV0 LDA@#FF;STA#B803
530 LDA@0;STA#91;STA#92;STA#93;STA#96;STA#97
540 LDA@#84;STA#94;JSR#FE6B;RTS
550\*** WRITE ADRESS ***
560:VV1 LDA@#FF;STA#B802;LDA#91;STA#B801;LDA#92;STA#B800
570 LDA@#CE;STA#B80C;LDA@#CC;STA#B80C;RTS
```

```
580\*** INCREMENT ADRESS ***
590:VV2 INC#93;BNEVV3;INC#94
600:VV3 INC#91;BNEVV4;INC#92
610:VV4 LDA@#10;AND#92;RTS
620\*** READ BYTE ***
630:VV5 LDA@0;STA#B802;LDA@#EC;STA#B80C
640 LDA#B800;STA#95;LDA@#CC;STA#B80C;RTS
650\*** P."ERROR" ***
660:VV6 JSR#F7D1;];?P=#07;$P+1="ERROR ";P=P+7;[;NOP;RTS
670\*** P."BLANK CHECK" ***
680:VV15 JSR#F7D1;];$P="BLANK CHECK ";P=P+12;[;NOP;RTS
690\*** P."OK" ***
700:VV16 JSR#F7D1;];$P="OK";P?2=#0D;P?3=#0A;P=P+4;[;NOP;RTS
710\*** P."ON #"(ADRESS) ***
720:VV17 JSR#F7D1;];$P="ON #";P=P+4;[;NOP
730 LDA#92;JSR#F802;LDA#91;JSR#F802;JSR#CD54;RTS
740\*** P."VERIFY " ***
750:VV24 JSR#F7D1;];$P="VERIFY ";P=P+7;[;NOP;RTS
760\*** P."READY" ***
770:VV20 JSR#F7D1;];$P="READY";P?5=#0D;P?6=#0A;P=P+7;[;NOP;RTS
780\*** P."SUM=#"(SUM) ***
790:VV25 JSR#F7D1;];$P="SUM=#";P=P+5;[;NOP
800 LDA#97;JSR#F802;LDA#96;JSR#F802;JSR#CD54;RTS
810\*** P."SET program" ***
820:VV31 JSR#F7D1;];$P="SET program";P?11=7;P=P+12;[
830 NOP;RTS
840\*** P."SET read" ***
850:VV34 JSR#F7D1;];$P="SET read";P?8=7;P=P+9;[
860 NOP;RTS
870\*** P."PROGRAMMING" ***
880:VV35 JSR#F7D1;];$P="PROGRAMMING";P=P+11;[;NOP;RTS
890];N.;@=4
900 P.$6"BEGINADRES=#"&D'"EINDADRES=#"&P'
910 P."LENGTE="P-D" BYTES"'
920 @=8;E.
>
>
```

TOELICHTING

```
           INITIALISATIE ROUTINES:
100        verwijder printer driver
110        zet 6522 klaar
120 - 160  display tekst en wacht op toets
           INLEES ROUTINES:
180 - 190  lees commando en kijk naar Q (Quit)
200        reset 6522 en printer driver, terug naar BASIC
210 - 220  interpreteer commando en spring naar routine
230        onbekend commando, display "ERROR"
240        hulp JMP voor conditionele sprong )256
           HOOFD ROUTINES:
260 - 290  blank check routine; commando (B)
310 - 330  read (p)rom routine; commando (R)
350 - 380  checksum berekenen ; commando (S)
400 - 440  verify prom routine; commando (V)
460 - 500  write prom routine ; commando (W)
           SUBROUTINES:
520 - 540  zet zero-page #91 - #97 klaar
560 - 570  schrijf prom adres (#91,#92) naar programmer
590 - 610  incr. ram (#93,#94) en prom (#91,#92) adres
630 - 640  lees byte van programmer in #95
660 - 880  tekst output routines
```

DERDE GRAADSVERGELIJKING

Dit is speciaal voor wiskunde maniakken . Velen kennen wellicht de oplossingsmetode voor een kwadratische vergelijking . Voor een derde graads bestaat er ook een metode ; hiervoor is speciaal het programma 3graad geschreven . Voor de liefhebbers: ook vierde graads vgl. kunnen exact opgelost worden (vraag mij niet hoe), men kan echter wiskundig bewijzen dat vgl. met een hogere graad dan vier niet meer exact op te lossen zijn.Hiervoor moet men zich beroepen op benaderingen.

Goed,beginnen we bij het begin:

$A*X^3+B*X^2+C*X+D=0$;onderstellen we $A\neq0$ dan kunnen we delen door A zodat mits nieuwe definities voor A,B,C,D:

$X^3+B*X^2+C*X+D=0$;hierin doen we de substitutie $X=Z-B/3$ zodat:

$Z^3+P*Z+Q=0$ met $P=C-B^2/3$ en $Q=2*B^3/27-B*C/3+D$.

Een tweede substitutie (de Viëta substitutie) $Z=W-P/(3*W)$; hiervoor is:

$(W^3)^2+Q*W^3-P^3/27=0$;dit is een tweede graadsvgl. in W^3, dus:

$W^3=-Q/2+(Q^2/4+P^3/27)^{1/2}$ en $W^3=-Q/2-(Q^2/4+P^3/27)^{1/2}$

Dus in principe 2 oplossingen voor de derde macht van W.Aangezien een getal nu steeds 3 derde machts wortels heeft (waaronder minstens 2 complex) vinden we 6 oplossingen voor W ; op het eerste gezicht zijn er dus ook 6 voor Z,waarvoor we er echter maar 3 verwachten . Nu blijkt dat (enkele testen van mij bevestigen dit;dus 't zal wel waar zijn) de Z waarden bekomen uit die voor de derde macht van W in het eerste geval gelijk zijn aan die voor het tweede geval.Dit is een gevolg van het verband tussen Z en W.Om ook het geval P=0 in te sluiten zullen we de tweede oplossing voor W^3 nemen.

Hoe trekken we nu nog de derde machts wortel? Wel stel $W^3=X+iY$ met i de imaginaire eenheid. Noem R de norm en T de fase van W^3 dan is dus

$R=(X^2+Y^2)^{1/2}$ en $T=\arccos(X/R)$ als $Y\geq0$ en
$T=2*PI-\arccos(X/R)$ als $Y<0$ .Hiermee zijn de drie waarden voor W gegeven door:

$$W=R^{1/3}(\cos((T+2*PI*I)/3)+i*\sin((T+2*PI*I)/3) \quad I=0,1,2$$

Hieruit dus drie waarden voor W . Dit is echter niet de oplossing!! We moeten via Z nog X berekenen . Dit is kinderspel.

Nu het programma;het programma loopt iets anders af dan hier geschetst.Dit is een gevolg van het feit dat sommige stappen in een computer veel beter op een kortere manier gedaan worden.Maar toch, de draad zit erin,en het moet niet moeilijk zijn de werking te herkennen.

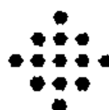
Programma

```
 10 REM 3DE-GRAADS VGL!!!!!
 20 REM ****************
 30 REM *cuypers frank***
 40 REM *gagelweg 50*****
 50 REM *B-3583 overpelt*
 60 REM **011/647601*****
 70 REM ****************
 80 P.$12
 90 P."3DE-GRAADS VGL!!"'
100 P."****************"''
110 P."OPLOSSEN VAN :"''
120 P."A*X↑3+B*X↑2+C*X+D=0"''
130 P."VOOR A,B,C,D REEEL."'
140 P."A<>O WORDT ONDERSTELD!!"''
150aDO FINPUT"A"%A;FUNTIL %A<>O
160 FINPUT"B"%B,"C"%C,"D"%D
170 %B=%B/%A,%C=%C/%A;%D=%D/%A
180 %P=%C-%B*%B/3
```

```
190 %Q=2/27*%B*%B*%B-%B*%C/3+%D
200 FIF %P*%Q=0;FP.'"ENIGE OPLOSSING:"'-%B/3';GOTOc
210 %E=%Q*%Q/4+%P*%P*%P/27
220 FIF %E>=0 %Y=0;%X=-%Q/2-SQR%E
230 FIF %E<0 %X=-%Q/2;%Y=-SQR(-%E)
240 %R=(%X*%X+%Y*%Y)
250 %S=SQR%R
260 FIF %R>0 %R=%R^(1/6)
270 P=(%X=0 AND %Y=0)
280 IF P %T=0;GOTOb
290 FIF %X=0 %T=PI/2+(SGN%Y-1)*PI/(-2);GOTOb
300 FIF %Y>=0 %T=ACS(%X/%S)
310 FIF %Y<0 %T=PI+ACS(%X/%S)
320bP.''" DE OPLOSSINGEN ZIJN:"''
330 ∂=0
340 FOR I=0 TO 2
350 %U=%R*COS((%T+2*I*PI)/3);%V=%R*SIN((%T+2*I*PI)/3)
360 %W=%P/3/(%U*%U+%V*%V)
370 FP.%U-%W*%U-%B/3,"+i*",%V+%W*%V'
380 NEXT I
390cP.''"ZO,OP NAAR DE VOLGENDE!"''
400 GOTOa
```

Een voorbeeldje: geef voor a,b,c,d resp. in 1,-6,11,-6 dan zijn de oplossingen (controleer!!) 1,2 en 3.De computer zal antwoorden met:
2.00000000+i*-6.98491931E-10
1.00000000+i*3.49245965E-10 en
3.00000000+i*3.49245965E-10
Wegens afrondigsfouten bij het berekenen van SIN , COS en ACS is het imaginair deel niet =0 . De kleine waarde ervan (E-10!!!) geeft ons de zekerheid dat ze nul moet zijn.

CUYPERS FRANK



Vervolg van het programma "INFOBLOK" van pagina 15.

```
155$q=$n
160doq=q+Lenq+1:in.$q:u.?q=13
170p?254=13:p?255=13
180!#a8=p:!#aa=0:!#ac=1
185p."opn":Li.#ffe3
190Li.((?18+3)*256+3)
200!#a8=b:!#aa=s:!#ac=s+(e-b)/256+1
210f.q=1to10:wait:n.
220Li.((?18+3)*256+3):e.
300bdo!#a8=p:!#aa=0:!#ac=1
310Li.((?18+3)*256+L)
320p.$(p+16)'
330u.$n=$(p+16):!#a8=b:!#aa=12!p:!#ac=12!p+(4!p-!p)/256+1
340Li.((?18+3)*256+L):e.
```

WAAR OF NIET WAAR?

(voorkennis: par. 2.8,4.8,4.9 en 8.2.5 uit Atom handboek)

Dit verhaaltje probeert enige duidelijkheid te verschaffen in het gebruik van de zgn. logische expressie d.w.z. een expressie die na evaluatie ofwel TRUE (waar) is, ofwel FALSE (onwaar).
De rekenprecisie van logische operaties is vier bytes, waarvan ter bepaling van de waarde TRUE of FALSE alleen het laagste byte wordt beschouwd.
In de Atom wordt de waarde FALSE gerepresenteerd door 0 en de waarde TRUE door een getal ongelijk aan 0. Na evaluatie van een logische expressie wordt de waarde TRUE echter gerepresenteerd door het getal 1.
Met deze kennis en die uit het Atom handboek vallen er over het gebruik van de logische expressie de volgende opmerkingen te maken.

1. De constructies IF A<>0 THEN... en IF A THEN... zijn niet identiek. Als A ongelijk aan nul is dan levert de eerste test altijd de waarde TRUE op. In het tweede geval wordt het low byte van A beschouwd en dat is niet alleen 0 als A gelijk aan 0 is, maar ook als A=256. Bewijs:
A=0;DO DO A=A+1;U.A;P.A;U.0 256,512,768 enz. worden niet afgedrukt; wel bij: A=0;DO DO A=A+1;U.A<>0;P.A;U.0

2. De tekens &, \ en : zijn specifiek bedoeld om bitoperaties uit te voeren op getallen. De "connectives" AND en OR zijn bestemd om logische expressies te koppelen. Daarom bestaat er ook geen EXOR in onze Basic; deze wordt bijna nooit gebruikt. Aangezien de logische en rekenkundige expressies nauw verweven zijn in Atom Basic kun je de connectives wel door elkaar gebruiken, maar netjes is dat niet.
Voorbeeld van het principiële verschil tussen & en AND:
IF ?#B001&#80=0 AND ?#B002&#40=0 THEN P."shift en repeat ingedrukt"

3. Het is jammer dat de & en AND intern op dezelfde manier worden afgehandeld, want dat is er de oorzaak van dat AND en OR niet in combinatie met FIF en FUNTIL gebruikt mogen worden.
De AND is eenvoudig te omzeilen:
FIF %A>2.5 FIF %A<9. THEN... en
DO DO ....;FUNTIL %A=%B;FUNTIL %A>12.5
zijn equivalent met de niet bestaande constructies:
FIF %A>2.5 AND %A<9. THEN... en
DO ....;FUNTIL %A=%B AND %A>12.5
De OR is veel moeilijker; het programma zal dusdanig ge- of herschreven moeten worden dat OR niet nodig is.

4. Vele Basics kennen de NOT-operator. Wij kennen die niet, maar hebben die ook niet nodig. Ten eerste heeft bijna elke logische operator een complement: NOT(A>2) is hetzelfde als A<=2, maar bovendien is NOT(A>2) hetzelfde als (A>2):1
Dus NOT(expr.) schrijven wij als (expr.):1

5. Als we de variabele A als een logische variabele beschouwen d.w.z. A kan alleen maar 0 of 1 zijn, dan kunnen we de NOT ook op deze manier verwezelijken: A=(A=0) i.p.v. A=NOT A of A=A:1
Dus A wisselt hier van waarde.

6. De tabel voor de EQUIVALENCE-operator EQV ziet eruit als:

A	B	(A EQV B)	(A EXOR B)	(NOT(A EXOR B))
0	0	1	0	1
0	1	0	1	0
1	0	0	1	0
1	1	1	0	1

Dus: A EQV B is hetzelfde als (A:B):1

7. Logische expressies kunnen ook strings bevatten, maar dan is alleen de =-operator toegestaan. De <>-operator kunnen we, gezien punt 4, omzeilen:
IF $A<>$B THEN... wordt IF ($A=$B):1 THEN...
Als we strings willen sorteren, zouden we ook constructies moeten hebben als:
IF $A>$B THEN...
Wie hier een elegante oplossing voor heeft, mag het zeggen!

IF (u begrijpt bovenstaand stukje):1 THEN neem contact op met

Bram Poot.

P.S. Bij de uitleg van de EQUIVALENCE-operator (p.2.27) moet nog worden toegevoegd dat A en B daar beschouwd moeten worden als logische expressies en niet als getallen. A=0 EQV B=0 schrijven wij als ((A=0):(B=0)):1. Willen we een EQV op bitniveau dan moeten we schrijven (en A en B zijn nu wel getallen): (A:B):#FFFFFFFF of (A:B):-1.
Bij AND en & wordt gesteld dat deze identiek zijn, maar dat is niet helemaal waar. AND neemt nl. alleen het low byte van de volgende expressie en is daarmee niet commutatief waar & dat wel is (een operator ! is commutatief als A!B=B!A). Kijk maar eens goed naar deze uitkomsten en let op de verschillen:
A=#0055AAFF;B=#0000F055
PRINT &(A AND B),&(B AND A),&(A & B),&(B & A) geeft:
0055AA55 0000F055 0000A055 0000A055

```
LIST   **BOOLE FORMULES IN BASIC**
    0 REM **BOOLE FORMULES IN BASIC
    1 REM **BRON: MINI MICRO JUNI '82
    2 REM **BEWERKT VOOR DE ACORN ATOM
    3 REM **DOOR P.H. VAN MOURIK
    4 REM **RUITERSTEDE 60, NIEUWEGEIN
    5 REM **GEBRUIKT PRINTER (MICROLINE) EN F.P. ROM
    6 REM **PRINT 1-8 VARIABELEN (A T/M H)
    7 REM **EN 1-8 FUNKTIES (P T/M W); DEZE WORDEN IN PROGRAMMARE-
    8 REM **GELS OPGENOMEN VANAF REGEL 1210 (HET ZIJN SUBROUTINES)
    9 REM
   80 DIM Z(8);?Z=0;Z!1=0;Z!5=0
   90 P.$12"1- PRINT TABEL"'"2- PRINT OPGEGEVEN VARIABELEN"';IN.N
  100 H=0
  110 IF N=1;IN."HOEVEEL VARIABELEN"N;G.170
  120 H=0
  130 IN."TIK AANTAL VARIABELEN",N,"HEX WAARDE DER VARIABELEN"H
  140 FIF H )= 2↑N;P.'"KAN NIET"';G.100
  150 I=0;L=H
  160 DO I=I+1;Z?I=H%2;H=H/2;U.H=0;H=L
  170 IN."MAX.8 FUNKTIES"M
  180 GOS.a;P.$27$66$12;?#FE=0
  190 P.$10;@=3;P."NR.  "
  220 F.I=1TO N;P."   "$I+64;N.;P."   !   "
  230 F.I=1TO M;P.$79+I"   ";N.;P.'
  240 P."           "
  250 F.I=1TO N;P."---";N.;P."!--"
  260 F.I=2TO M;P."---";N.;P."-"'
  270 L=H
  280 Y=0
  290 DO GOS.x
  300 IF Y=2;G.420
  310 P.L"  "
  320 F.I=N TO1 S.-1;P.Z?I;N.;P."   !"
  330 F.I=1TO N
  340 GOS.(1100+I*10)
  350 N.
  360 F.I=1TO M
  370 GOS.(1200+I*10)
  380 N.
  390 P.'
  400 IF H;Y=2
  410 L=L+1
  420 U.Y=2
  430 P.$10$10"GEBRUIKTE FUNKTIES ZIJN:"'
  435 P.$3;P."(MOMENTJE)"';GOS.a
  440 I=(?#12)*#FF;DO I=I+1;U.?I=4 AND I?1=#BA;I=I+2
  445 DO
  447 IF I?1()61;G.450
  449 P.$I'
  450 I=I+L.I+4;U.$I="END"
  460 P.$3$6;END
  470xIF Y=0;Y=1;R.
  480 X=1
  490 IF X)N;Y=2
  500 IF Z?X=1;X=X+1;G.490
  510 Z?X=1
  520 F.I=X-1 TO 1 S.-1;Z?I=0;N.
  530 R.
```

```
 600aREM INDIEN GEEN PRINTER:
 610 REM      REGEL 620 RETURN
 620 P.$2 ; RETURN
1110 A=Z?(N-0);R.
1120 B=Z?(N-1);R.
1130 C=Z?(N-2);R.
1140 D=Z?(N-3);R.
1150 E=Z?(N-4);R.
1160 F=Z?(N-5);R.
1170 G=Z?(N-6);R.
1180 H=Z?(N-7);R.
1210 P=A&B;REM AND FUNKTIE
1215 P.P&1;R.
1220 Q=A&B:1;REM NAND
1225 P.Q&1;R.
1230 R=A|B;REM OR
1235 P.R&1;R.
1240 S=A|B:1;REM NOR
1245 P.S&1;R.
1250 T=A:B;REM EOR
1255 P.T&1;R.
1260 U=A:1;REM NOT A
1265 P.U&1;R.
1270 V=A|B&(C:1)|C&((D:1)|(A:1))
1275 P.V&1;R.
1280 W=A&B
1285 P.W&1;R.
1300 END
```



```
)1- PRINT TABEL
2- PRINT OPGEGEVEN VARIABELEN
?1
HOEVEEL VARIABELEN?2
MAX.8 FUNKTIES?8
```

NR.	A	B	!	P	Q	R	S	T	U	V	W
0	0	0	!	0	1	0	1	0	1	0	0
1	0	1	!	0	1	1	0	1	1	1	0
2	1	0	!	0	1	1	0	1	0	1	0
3	1	1	!	1	0	1	0	0	0	1	1

```
GEBRUIKTE FUNKTIES ZIJN:
Q=A&B:1;REM NAND
R=A|B;REM OR
S=A|B:1;REM NOR
T=A:B;REM EOR
U=A:1;REM NOT A
V=A|B&(C:1)|C&((D:1)|(A:1))
W=A&B
```

toelichting SCREENDUMP

Dit programma produceert een afdruk van het CLEAR 4 scherm, naar keuze groot of klein, op een printer.(een STAR DP 510)
Het bijzondere aan dit programma is echter dat het te gebruiken is TIJDENS het draaien van een ander programma.

Het idee is als volgt:

* het eerste deel van het programma programmeert de VIA (die dan natuurlijk wel aanwezig moet zijn !!) zo, dat deze op gezette tijden een bepaalde routine aanroept, waarvoor hij eerst de werking van het hoofdprogramma stopzet.

* In die routine wordt gekeken of er bepaalde toetsen zijn ingedrukt (namelijk CTRL-SHIFT voor een grote en CTRL-REPT voor een kleine afdruk)

* Is dit niet het geval dan wordt gewoon teruggesprongen naar het hoofdprogramma

* Zijn de bewuste toetsen WEL ingedrukt, dan wordt eerst de bewuste afbeelding naar buiten gestuurd, waarna weer doodleuk naar het hoofdprogramma wordt gesprongen

Doordat de "toetsleesroutine" erg kort is, is normaalgesproken niet te merken dat het programma (Passief) aanwezig is. Als de routine echter in het hoge (gestapelde) gebied (#9800 e.v.) staat, dan wandelt er continue een streep over het beeld (is wel makkelijk om te zien of je de routine er niet per ongeluk uit "ge-BREAKed" hebt)

Als het programma gebruikt wordt met andere programma's die (veel) ZERO-page adressen gebruiken (b.v. (machinetaal-) spelletjes) kan dat wel eens problemen geven. Dit is de reden waarom nogal wat incourante zeropage-adressen worden gebruikt (nl. die van de DISC)

------------------------------------ succes ------------------------------------

Andre de BRUIN
Baljuw 11
2671 HL NAALDWIJK

```
>LIST

  10 P.$12"    *** screen-copy ***"
  20 P.''"DIT PROGRAMMA PRODUCEERT EEN"
  30 P.'"HARD-COPY VAN EEN CLEAR4 SCHERM."
  40 P.'"DIT WORDT BEREIKT DOOR EEN "'
  50 P."INTERUPTPROGRAMMA. DIT BETEKENT"'
  60 P."DAT HET PROGRAMMA AAN TE ROEPEN IS TIJDENS"
  70 P." DE UITVOERING VAN"'"EEN ANDER PROGRAMMA"'
  80 P."BEDIENING:"''"COPY (KLEIN)      COPY(GROOT)"'
  90 P."ctrlrept                ctrlshift"'
 100 P.'"GEBRUIKTE ZEROPAGE: #B0..#BF(!)"
 110REM #B0..#BF IS EIGENLIJK DISC-WERKRUIMTE !!!!!
 120 LINK#FFE3;P.$12''''
 130 IN."WAAR CODE ,(BIJV #9A00) "W
 140 DIM VV30
 150 DIM KK30
 160 DIM LL10
 170 F.J=1TO30;VVJ=W;KKJ=W;LLJ=W;N.
 180 P.$21
 190 FORI=1TO2;P=W
 200 B=KK2%256;C=KK2/256
 210[
 220:KK0  \    {INTERUPTROUTINE }
 230 LDA@#40;STA#B80B;LDA@#C0;STA#B80E;LDA@#50;STA#B804;LDA@#FF
 240 STA#B807;STA#B805;LDA@B;STA#204;LDA@C;STA#205
 250 CLI;RTS
 260\   { DE VIA IS NU ZO GEPROGRAMMEERD DAT HIJ OP GEZETTE
 270\       TIJDEN HET PROGRAMMA "KK2" AANROEPT }
 280 :KK2;LDA#B804;STY#BB;STX#BC{ SAVEN VAN X & Y }
 290 LDA#B001;CMP@63;BEQ KK4        { IS ctrlshift INGEDRUKT ? }
 300 LDA#B001;CMP@191;BEQ KK3       { IS ctrl INGEDRUKT ? }
 310 JMPVV10                        { TERUG NAAR HOOFDPROGRAMMA }
 320 :KK3
 330 LDA#B002;AND@64;BEQ KK7        { DAARBIJ OOK NOG DE rept ? }
 340 JMP VV10
 350:KK7
 360 LDA@0;STA#B0;JMP KK5
 370:KK4
 380 LDA@1;STA#B0
 390:KK5
 400 LDX@0;LDY@0                    { INITIALIZATIE }
 410 LDA@0;STA#B1;LDA@#80;STA#B2
 420 LDA#B0;CMP@1;BEQLL1
 430 LDA@#C0;STA#BD;LDA@#A0;STA#BE;LDA@#4B;STA #B6
 440 LDA@#20;STA#BF;LDA@1;STA#B4
 450 JMP LL2
 460:LL1
 470 LDA@#60;STA#BD;LDA@#40;STA#BE;LDA@#4C; STA #B6
 480 LDA@#40;STA#BF;LDA@3;STA#B4
 490:LL2
 500 LDA@2;JSR#FEFB                 { PRINTER AAN }
 510 LDA@#1B;JSR#FEFB               { REGELAFSTAND OP 6/72 INCH }
 520 LDA@#41;JSR#FEFB
 530 LDA@6;JSR #FEFB
```

```
540\
550\
560:VV6
570 LDA@0;STA#B7
580 LDA@#1B;JSR #FEFB;LDA #B6 ;JSR #FEFB; LDA@0; JSR #FEFB
590\        (INSTELLEN RESOLUTIE V/D PRINTER)
600 LDA #B4;JSR VV25
610 :VV5
620  LDA@128; STA#B3
630   :VV4;LDA@0; STA#B5; LDY@0
640    :VV3
650     LDA(#B1),Y;AND#B3; SEC; BNE VV1
660     CLC; :VV1; ROL#B5; LDA#B0; CMP@1; BNE VV2
670     ROR#B5; BCC VV11
680              ROL#B5; SEC;  ROL#B5;  JMPVV12
690     :VV11; ROL#B5; CLC; ROL#B5
700     :VV12
710     :VV2; TYA; CLC; ADC@#20; TAY;   CMP#BD
720    BNE VV3
730    LDA#B5; JSR VV25
740    LDA#B0; CMP@1; BNE VV7
750    LDA#B5; JSR VV25; LDA#B5; JSR VV25
760    :VV7; CLC; ROR#B3
770   BCC VV4
780   CLC; LDA#B1; ADC@1; STA#B1; LDA#B2; ADC@0
790   STA#B2;  CLC;  LDA#B7;  ADC@1;  STA#B7
800   CMP@#20
810 BNE VV5
820 LDA@#0D;JSR#FEFB                              ( EINDE 1 REGEL )
830 CLC; LDA#B1; ADC#BE; STA#B1; LDA#B2
840 ADC@0; STA#B2; INX; CPX#BF;BEQ VV16
850JMP VV6
860\
870\
880:VV16
890 LDA@#1B;JSR#FEFB;LDA@#40;JSR#FEFB       ( PRINTER NORMAAL )
900LDA@3;JSR#FEFB                           ( PRINTER UIT )
910:VV10
920 LDX#BC;LDY#BB;PLA;RTI       ( TERUGKEREN UIT DE INTERUPT )
930\
940\
950\    ( :VV25 IS EEN NAGEMAAKTE #FEFB, DIE ECHTER
960\        NIET CONTROLEERT OP #2,#3 OF #FE   )
969:VV25 PHA;JMP #FF08
1030]
1040 N.;P.$6
1050 LINK W
1060 P.$12''"PROGRAMMA IS NU (PASSIEF)"
1070 P.'"AANWEZIG, AANROEP DOOR"
1080 P.'" ctrlshift   OF ctrlrept "''
1090 P."NA EEN break WEER OPSTARTEN ";@=0
1100 P.'"MET  LINK #"&W
1110 END
```


10
amsterdam 25 april 1968
SECRETARIS PRESIDENT
TIEN GULDEN 10




10
amsterdam 25 april 1968
SECRETARIS PRESIDENT
TIEN GULDEN 10





Beste mensen,

om te beginnen een rectificatie.
In mijn programma 'FORMAT' (Acorn Nieuws nr.4) zitten drie foutjes: in regel 270 moet LL6 vervangen worden door LL5, in regel 280 BPL door BCS en in regel 330 BMI door BCC.
Met de gepubliceerde versie van het programma kan het voorkomen dat een regel niet op de printer verschijnt. Onaanvaardbaar natuurlijk.
Ten tweede: een onduidelijkheid.
Kondig ik in hetzelfde artikel het programma 'OPTILLUSIE' aan, is er nergens een listing te vinden. *Ja!*
Ben ik vergeten een listing mee te sturen of is de redactie (Ton, weet jij het nog ?) vergeten deze te plaatsen ? Bij deze alsnog een listing.
Zo, dat is weer rechtgezet. *Inderdaad!* *Nee!*

Van mijn A-D/D-A converter heb ik tot nog toe geen resultaten laten zien. Dat wordt dan wel hoog tijd.
Op pag. zien we een printout van enkele golfvormen. De bemonsteringsfrequentie bedraagt 21kHz.Deze beelden zijn geproduceerd via mijn programma 'DIGIPLOT'.
Dit programma verwacht een stuk geheugen van #0000 tot #9FFF (minder is nauwelijks zinvol: met deze snelheid zijn de voor data-opslag bestemde 28kBytes in een goede seconde vol) en biedt de volgende faciliteiten:

* inlezen van een analoog signaal
* herhaald weergeven van dit signaal
* herhaald weergeven van een deel van dit signaal (2bytes tot 28kBytes)
* uittekenen van het signaal op het scherm,waarbij de tijdas naar believen kan worden uitgerekt en ingekrompen
* aansluitend dumpen van het getekende signaal naar een printer

Met dit gereedschap valt een aardige signaalanalyse te plegen.
Mochten er meer mensen zijn met een 'vol' geheugen en een (plan tot bouwen van) een converter (bestaat er enig inzicht bij de club w.b. de diverse uitbreidingen aan onze rommelcomputer?) dan wil ik wel een kopie naar het bandjesarchief zenden.

Mijn stokpaardje: GRAPHICS !

256*192 punten oplossend vermogen is voor een goedkope micro al behoorlijk, maar ik vond het niet genoeg.
Op weg naar 512*768 punten ben ik voorlopig blijven steken bij 512*384 dots. Ook niet kinderachtig want zo'n plaatje slurpt al 24kByte op.
Hiertoe heb ik het geheugengebied van #2000 tot #8000 bestempeld tot quasi-grafisch geheugen, d.w.z. je brouwt hier plaatjes, maar in principe zie je hier niets van op je scherm.
Omdat nu die 24 kBytes teveel zijn voor het video-geheugen (6kB max.) maar het toch wenselijk is om te kunnen zien wat er in staat heb ik via het programma een venster van 6kB (jawel, het video-geheugen) gemaakt op de grote plaat.
Dit 'venster' is voor te stellen als een passe-partout dat de onderliggende afbeelding ruimschoots bedekt en waarin in het midden een rechthoekig gat is gemaakt ter grootte 256*192 punten. Dit passe-partout nu is in 4 richtingen te verschuiven zodat ieder punt op de afbeelding op het scherm zichtbaar te maken is (maar niet ieder paar punten tegelijk). Dit principe noemt men 'windowing'.
Dit alles heeft als beperking dat niet de gehele afbeelding in een keer op het scherm verschijnt, maar steeds slechts een vierde deel hiervan. Er zijn echter meerdere voordelen, zoals het kunnen vermenigvuldigen van beeldgedeeltes, het verkleinen van de gehele afbeelding naar het scherm zonder het origineel in eerste instantie aan te tasten en diverse andere trucs die met woorden nauwelijks te beschrijven zijn.

Bijbehorend is een programma met de volgende mogelijkheden:
* d.m.v. 4 toetsen is het venster (window) in stapjes van 8 dots in 4 richtingen te verplaatsen
* tekenen van lijnen in 8 richtingen (zie 'DRAW'/Hobbit)met een snelheid van naar keuze 10 of 60 dots per seconde
* invoeren van tekst
* inverteren van het plaatje
* wissen van het plaatje
* laden van een 'groot' plaatje (24kB.)
* saven van een 'groot' plaatje (24kB.)
* laden van een 'klein' plaatje (display: 6kB.)
* saven van een 'klein' plaatje (display: 6kB.)
* aanbrengen van een raster over het plaatje (en weer verwijderen)
* aanbrengen van een kader rond het plaatje (en weer verwijderen)
* het vergroten van het op het display zichtbare gedeelte van het plaatje naar het gehele oppervlak (waarbij dus elk puntje vervangen wordt door een blokje van 4 puntjes)
* het verkleinen van de grote plaat naar het video-geheugen
* dumpen van het plaatje naar de printer
* diverse goochelfuncties m.b.v. waarvan beeldgedeeltes verhuisd/veranderd/gecombineerd kunnen worden

Ik heb geen assembler listing ingestuurd daar deze ondertussen een omvang heeft van ca.14kByte (en nog steeds groeit). Geassembleerd levert dit ongeveer 3,5kByte machinecode op.
Ons Atoompje is best wel snel eigenlijk: een two-pass assembly van deze lap vergt slechts 27 seconden !
Naar verwachting is het programma medio/eind oktober klaar en zal dan ook nog gesloten contouren in kunnen vullen met een stippenpatroon in een aantal dichtheden.

De tekeningen zijn geproduceerd door een gewoon BASIC-programma (iets dergelijks als op Atomic t&p, pag.83), met de standaard PLOT-opdrachten: ik heb alleen een machinetaalroutine hoeven schrijven die een puntje set/reset of inverteert in mijn quasi-grafisch geheugen en het adres van deze routine in te vullen op de adressen #3FE,#3FF.
Dank aan de ontwerpers van de Atom.

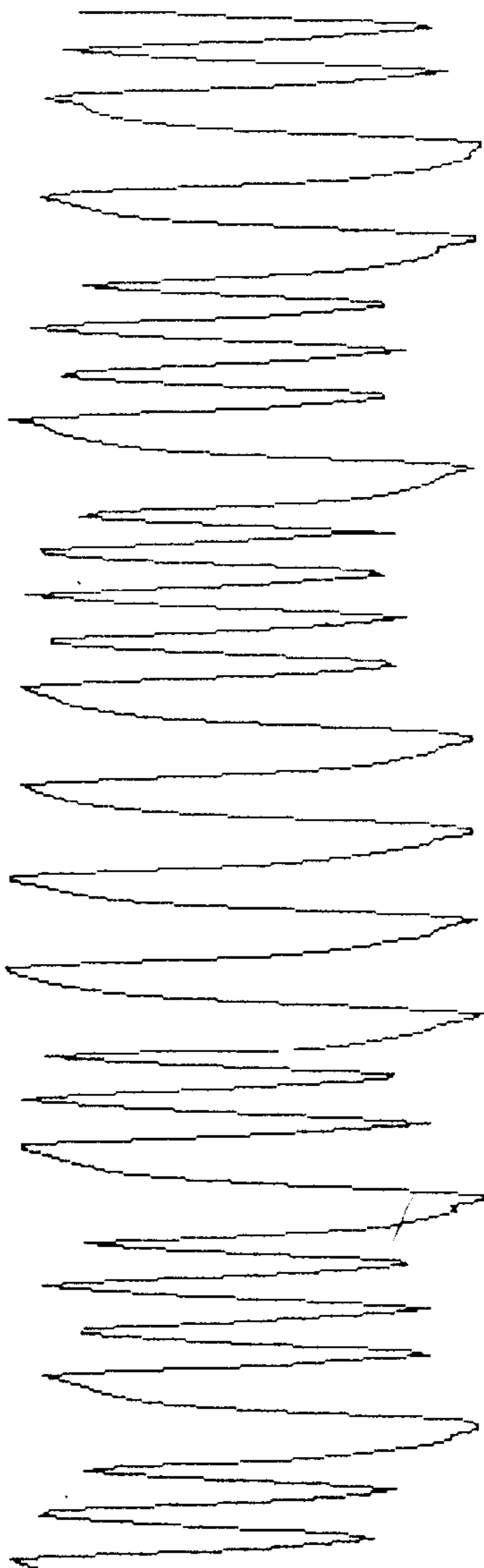
Het programma 'GLOBE' tenslotte tekent op uw 'kleine' scherm van 256*192 dots een kleine globe. Voordat het programma klaar is zijn 24 minuten verstreken !

Groetend ,

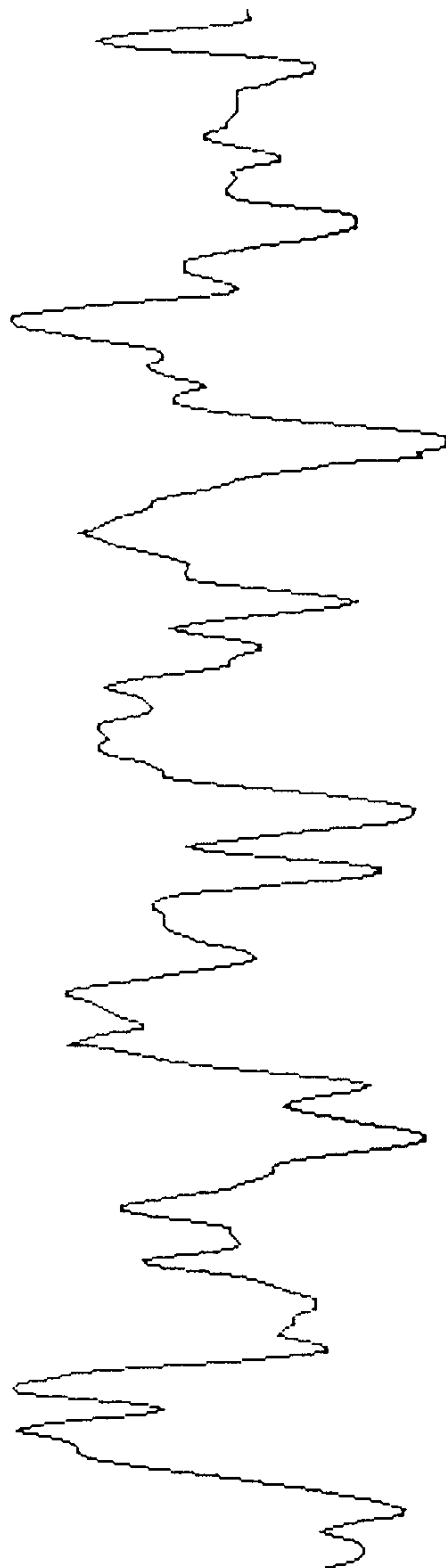
R. van

Rob van Dort

Zo legt de Atom zijn programma's vast (1200 Baud)



En zo ziet een stukje popmuziek eruit :



ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890!"#$%&'()-=:*@_\
÷→↓↑←;+[]<>,./?

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 0 ! " # $ % & ' ( ) - = : * @ _ \
÷ → ↓ ↑ ← ; + [ ] < > , . / ?

A B C D E F G H I J K L M N O P
a b c d e f g h i j k l m n o p
1 2 3 4 5 6 7 8 9 0 ! " # $ % &
÷ → ↓ ↑ ← ; + [ ] < > , . / ?

A B C D E F G H

A B C D E F G H

Demonstratie van enige mogelijkheden van 'SUPERDRAW'

*** GLOBE 14/9/'83 ***

```
  0     REM GLOBE 14/9/'83
 10     CLEAR4
 20     G=128;H=096;REM COORDINATEN MIDDELPUNT SCHERM
 30     K=8;REM FACTOR TER VERHOGING VAN DE PRECISIE (NIET K )8!!)
 40     L=100
 50     M=5
 60     N=30
 70     S=L*L+M*M;R=SQR S;T=S+N*N;Q=SQR T
 80     E=30*K;REM BOLSTRAAL
 90     REM TEKEN HORIZONTALE CIRKELS
100     FOR A=-90 TO 90 STEP 15
110       F=%(E*COS RAD A)
120       IF A=90 OR A=-90 THEN U=0;V=0;W=E;GOS.b;GOS.a;GOTO r
130       B=0
140       W=%(E*SIN RAD A)
150       U=%(F*COS RAD B)
160       V=%(F*SIN RAD B)
170       GOS.b
180       FOR B=2 TO 360 STEP 2
190         U=%(F*COS RAD B)
200         V=%(F*SIN RAD B)
210         GOS.a
220       NEXT B
230 r   NEXT A
240     REM TEKEN VERTIKALE CIRKELS
250     FOR A=-90 TO 90 STEP 15
260       W=0
270       U=%(E*COS RAD A)
280       V=%(E*SIN RAD A)
290       GOS.b
300       FOR B=2 TO 360 STEP 2
310         W=%(E*SIN RAD B)
320         V=%(E*SIN RAD A *COS RAD B)
330         U=%(E*COS RAD A *COS RAD B)
340         GOS.a
350       NEXT B
360     NEXT A
370     P.$7$7$7$7
[redacted]
390     END
400 a
410     I=U*L;J=V*M;O=R*(T*K-I-J-W*N)
420     C=T*(V*L-U*M)*4/O+G;D=H+3*Q*(W*S-N*(I+J))/O
430     PLOT5,C,D;R.
440 b
450     I=U*L;J=V*M;O=R*(T*K-I-J-W*N)
460     C=T*(V*L-U*M)*4/O+G;D=H+3*Q*(W*S-N*(I+J))/O
470     PLOT4,C,D;R.
```

PROGRAMMALENGTE 1033 BYTES

*** optillusie 14/7/'83 ***

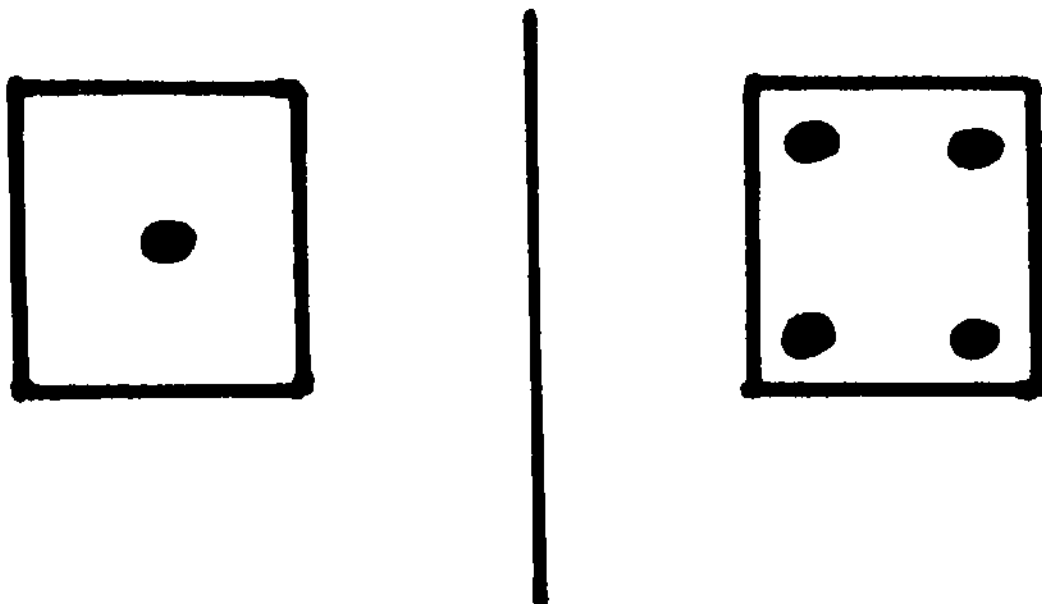
```
  0   REM optillusie 14/7/'83
 10   CLEAR4
 20   P.$21
 30   FOR I=0 TO 6000
 40   X=ABSRND %100
 50   Y=ABSRND %100
 60   PLOT13,X,Y
 70   IF X)15 IF X(85 IF Y)15 IF Y(85 THEN X=X-1
 80   IF X)30 IF X(70 IF Y)30 IF Y(70 THEN X=X-1
 90   IF X)45 IF X(55 IF Y)45 IF Y(55 THEN X=X-1
100   PLOT13,(255-X),Y
110   NEXT I
120   END
```

PROGRAMMALENGTE 277 BYTES

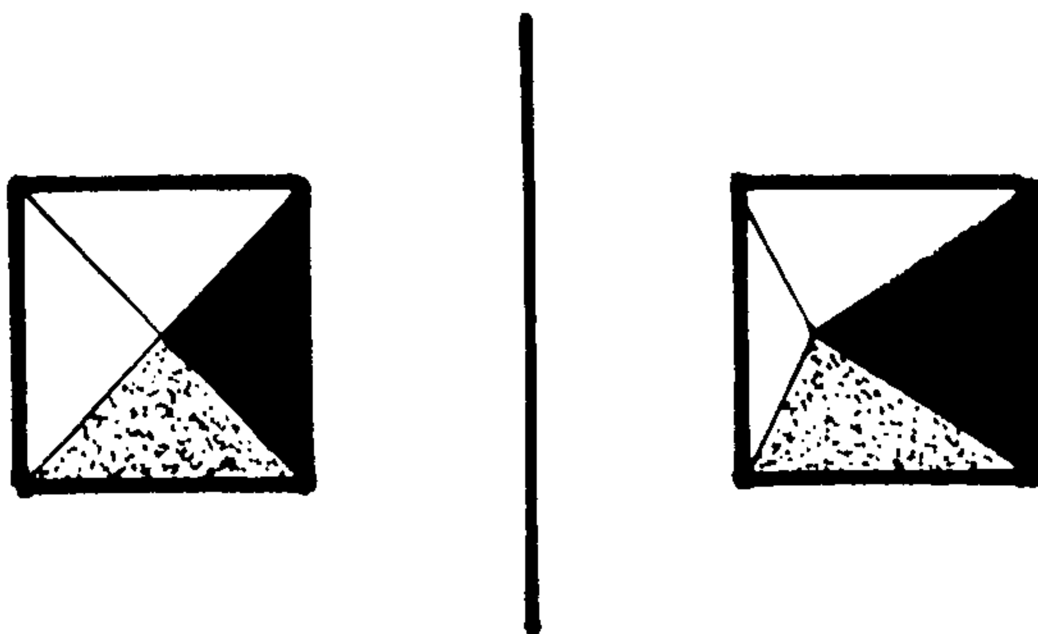
Gezichtsbedrog

Even inhaken op het gezichtsbedrog van Rob van Dort. Beelden in elkaar laten overvloeien gaat ook zonder een spiegel (ook al een oud kunstje).

Probeert u maar: zet een doodgewone briefkaart, ansichtkaart of een correspondentiekaart op de vertikale lijn. Zet uw neus op de kaart en de dobbelsteenfiguren 1 en 4 zullen samen 5 vormen (iets bewegen van het hoofd kan - net als bij de spiegel - noodzakelijk zijn).



Je kunt op deze manier ook iets driedimensionaal uitbeelden. Dezelfde methode en er verschijnt een pyramide met "diepte".



Ik neem aan (ik heb het niet geprobeerd)dat deze methode ook te gebruiken is op het beeldscherm (en uiteraard met een grotere kaart).

Overigens: met de computer kun je ook mooie gezichtsbedrog-figuren zichtbaar maken.

Voer het volgende programma in.

```
 1Ø P.Ø12"GEZICHTSBEDROG"''
 2Ø REM JBDEKKER
 3Ø P."DE BEIDE HORIZONTALE LIJNEN OP  "
 4Ø P."HET VOLGENDE PLAATJE LIJKEN KROM"
 5Ø P."MAAR ZIJN VOLKOMEN RECHT!"; LINK#FFE3; CLEAR4;
    A=128; B=96
 6Ø Y=192; FOR X= 384 TO -A STEP-32
 7Ø MOVE A,B; DRAW X,Y; N.X
 8Ø Y=Ø; FOR X=-A TO 384 STEP 32
 9Ø MOVE A,B; DRAW X,Y; N.X
1ØØ MOVE Ø,48; DRAW 256,48; MOVE Ø,144; DRAW 256,144; END
```

En zo is er nog veel meer te maken op het gebied van gezichtsbedrog. Ik heb er nog een paar in voorraad!

ZWART op WIT: er kan nòg meer met de Josbox

Voer het volgende ("probeer")-programma in.

```
1Ø DIM A16; C=#8ØØØ; P.Ø12" zwart op wit met de josbox"''
2Ø IN."VOER TEKST IN (MAX.16 KAR.-"'" GEINVERTEERD)"'ØA
3Ø P."WELKE LETTERGROOTTE?"''"1 GROOT"''"2 MIDDELMATIG"''
4Ø IN."3 KLEINER"''"4 NORMAAL"M; IF M=1 OR M=2 D=#86ØØ; G.7Ø
5Ø IF M=3 D=# 8CØØ; G.7Ø
6Ø IF M=4 D= #9800; G.7Ø
7Ø P.Ø12; GRMOD; CLEAR M; ?# E1=Ø
8Ø FOR X= C TO D STEP4; !X=-1; NEXT
9Ø P.ØA; LI.# FFE3; TXMOD; G.1Ø
```

J.B.DEKKER

Acorn Computerclub Overijssel/Gelderland

ONTWERP ZELF DE KARAKTERSET VAN JE MICROLINE 80

In de MICROLINE 80 printer zit standaard een karakter generator van het type HN462316. Met plug S1 in stand B en S3 in stand A worden de volgende karakters geprint:
- code #00 - #7F: 'standaard' ASCII karakters
- code #80 - #9F: grafische karakters
- code #A0 - #DF: Japanse tekens (KANA)
- code #E0 - #FF: grafische karakters.

Zoals reeds aangegeven in de handleiding van de printer kunnen we deze karakter-ROM vervangen door een I-2716 (INTEL fabrikaat, kosten ca. fl.15,-). Dit is een EPROM van 2K bytes die we zelf kunnen programmeren net als de 2532.
De bedoeling van dit alles is natuurlijk om de 64 Japanse tekens te vervangen door zinvollere karakters. Bovendien kunnen we de ASCII set in overeenstemming brengen met de karakterset van onze ACORN ATOM, d.w.z. de pijltjes die worden geprint voor code #5B - #5E vervangen door resp. [\ ] en ^ (je kunt zien dat dit op mijn machine al is gebeurd).

Voordat je je eigen karakterset kunt maken moet je natuurlijk eerst weten hoe de karakters zijn opgeslagen in de karakter ROM. Het heeft mij heel wat moeite gekost om dit uit te vinden. Met behulp van onderstaand programma kun je echter zelf karakters maken zonder te weten hoe ze worden opgeslagen! Het enige wat je hoeft te doen is het volgende:

1. Op de een of andere manier moet je de inhoud van de bestaande karakter ROM in het geheugen van je ACORN ATOM zien te krijgen. Dit kan op verschillende manieren, b.v. door het laden van de file "ML-ROM", het inlezen van de ROM via de programmer of via het tussenvoetje dat wordt gebruikt om 2732's te lezen.
Schrijf in ieder geval de inhoud van de HN462316 naar cassette!

2. Run het onderstaande programma.

3. Schrijf de zelfgemaakte karakterset weer naar cassette.

4. Stop de nieuwe karakterset in een 2716, of laat dit doen door de programmeerdienst van de club.

5. Vervang de HN462316-A00 van je microline door je eigen 2716, en kontroleer of de pluggen S1 en S3 goed staan.

Toelichting bij het programma.
Het programma gaat ervan uit dat de data van de karakter-ROM in het geheugen staat vanaf adres #6000. Indien nodig in regel 60 de **statement R=#6000 vervangen.**
Als je alles goed hebt ingetikt verschijnt er na het RUNnen een rooster van 5 bij 7 waarin je je eigen karakters kunt ontwerpen. Op de plaats die wordt aangegeven door een klein rechthoekje (de

cursor), kun je nu een stip laten verschijnen door de spatiebalk in te drukken. Nogmaals de spatiebalk indrukken en de stip verdwijnt weer. Door nu de cursor te verplaatsen kun je op elke gewenste plaats een stip laten verschijnen/uitvegen. Ben je helemaal tevreden met je eigen gewrocht, dan kun je dit karakter toe kennen aan een bepaalde toets, overeenkomend met een bepaalde hex code. Merk op dat er een stip geplaatst kan worden OP of TUSSEN de vertikale lijnen, resulterend in een dot-matrix van 9 bij 7 (zie fig.7 - 11 van je handleiding). Het is dan ook mogelijk dat de stippen elkaar overlappen. Ontwerp je karakters echter zodanig dat dit niet gebeurt!
Ook kun je een reeds bestaand karakter oproepen, eventueel veranderen en daarna aan een andere toets (hex code) toewijzen.
De vertaling van toets naar hex code geschiedt als volgt. Het programma kan in twee 'modes' verkeren: ASCII mode (A) en graphic mode (G). In ASCII mode selekteert het programma karakters, die overeen komen met hex kodes waarvan het hoogste bit Ø is, de codes ØØ-7F dus. Deze karakters komen overeen met het normale toetsenbord van je machine, d.w.z. leestekens, grote- en kleine letters. In GRAPHIC mode selekteer je de kodes waarvan het hoogste bit=1 is. Hierin bevinden zich momenteel de Japanse tekens. Als je b.v. in GRAPHIC mode het karakter ! oproept, dan krijg je het karakter waarvan de kode gelijk is aan die van het uitroepteken (#21) + #8Ø = #A1. Dit is een KANA karakter (rulletje in benedenhoek). Je kunt te allen tijde overschakelen tussen ASCII en GRAPHIC mode, behalve tussen een kommando dat uit twee toetsaanslagen bestaat. Het programma start in ASCII mode. Tenslotte de kommando's:

@ (toets)	haal het karakter op dat overeenkomt met (toets)
(return)	haal het karakter op waarvan de hex kode 1 hoger is (volgende karakter)
N	begin met de definitie van een nieuw karakter
Q	(quit) d.w.z. stop het programma
A	schakel over op de ASCII set (bit 7=Ø)
G	schakel over op de GRAPHIC set (bit 7=1)
\	cursor omhoog
/	cursor omlaag
[	cursor naar links
]	cursor naar rechts
C (toets)	ken het karakter toe aan een bepaalde toets (hex code)
(spatie)	teken een stip op de plaats van de cursor of haal deze weg (inverteer stip).

Verder zij nog opgemerkt dat het verstandig is om het spatie karakter toe te kennen aan alle ongebruikte codes, zodat deze in een later stadium kunnen worden ingevuld zonder de ROM te hoeven wissen!

```
>L.
   10 REM CHARACTER DESIGN FOR MICROLINE 80  5/08/83
   20 REM (C) MARTIN JANSSEN
   30 REM      BUITENSTEDE 30
   40 REM      3431 XB  NIEUWEGEIN
   50 K=#C0; Q=TOP;!Q=#85FFE320;Q!4=#60C0
   60 A=#41;R=#6000;B=R;D=2;M=0;GOS.a;X=0;Y=0;GOS.d
   70 DO; LINK Q
   80 IF?K=CH"@" GOS.a;LINK Q;A=?K;GOS.b;X=0;Y=0;GOS.d;U.0
   90 IF?K=#0D GOS.a;GOS.b;X=0;Y=0;GOS.d; A=A+1
  100 IF?K=CH"A" B=R; REM ASCII
  110 IF?K=CH"G" B=R+#400; REM GRAPHICS
  120 IF?K=CH"\" GOS.d;Y=Y+16;GOS.e;GOS.d
  130 IF?K=CH"/" GOS.d;Y=Y-16;GOS.e;GOS.d
  140 IF?K=CH"]" GOS.d;X=X+8; GOS.e;GOS.d
  150 IF?K=CH"[" GOS.d;X=X-8; GOS.e;GOS.d
  160 IF?K=CH" " GOS.c
  170 IF?K=CH"N" GOS.a;X=0;Y=0;GOS.d
  180 IF?K=CH"Q" P.$12; END
  190 IF?K=CH"C" LINK Q;A=?K;IF A>=#20 IF A<#80 GOS.f
  200 UNTIL 0
  210aCLEAR 4; J=-1; REM CLEAR SCHERM & TEKEN ROOSTER
  220 FOR I=#8000 TO #807E STEP 4; !I=J; N.; ?I=#0D
  230 FOR I=#8100 TO #9700 STEP #80; $I=$#8000; N.
  240 $#8081=$#8000; !#8080=J; !#8100=J
  250 FOR I=0 TO 5; X=I*16; MOVE X,0; PLOT 7,X,112; N.
  260 FOR I=0 TO 7; Y=I*16; MOVE 0,Y; PLOT 7,80,Y; N.; RETURN
  270bFOR I=0 TO 8; X=I*8; REM TEKEN KARACTER
  280 IF I<8 C=B?(#80*I+A)
  290 IF I=8 C=B?(A&#F0*8+#10+A&#F)
  300 Y=112; FOR J=0 TO 6; Y=Y-16
  310 IF C%2=0 GOS.c; REM DATA IN ROM IS INVERTED
  320 C=C/2; N.J; N.I; RETURN
  330cMOVE X,Y; REM TEKEN EEN PIXEL
  340 PLOT M, 6,1; PLOT D,  4,0; PLOT M, -6,1; PLOT D, 8,0
  350 PLOT M,-9,1; PLOT D, 10,0; PLOT M,-11,1; PLOT D,12,0
  360 PLOT M, 0,1; PLOT D,-12,0; PLOT M, -1,1; PLOT D,14,0
  370 PLOT M, 0,1; PLOT D,-14,0; PLOT M,  0,1; PLOT D,14,0
  380 PLOT M, 0,1; PLOT D,-14,0; PLOT M,  0,1; PLOT D,14,0
  390 PLOT M,-1,1; PLOT D,-12,0; PLOT M,  0,1; PLOT D,12,0
  400 PLOT M,-1,1; PLOT D,-10,0; PLOT M,  1,1; PLOT D, 8,0
  410 PLOT M,-2,1; PLOT D, -4,0; RETURN
  420dMOVE X,Y; REM TEKEN 'CURSOR' (INVERTEER)
  430 PLOT M,6,7; PLOT D,4,0; PLOT M,0,1; PLOT D,-4,0
  440 PLOT M,0,1; PLOT D,4,0; R.
  450eIF X<0 X=64;Y=Y-16;IF Y<0 Y=96; R.; REM TEST X EN Y
  460 IF Y<0 Y=96;X=X-8; IF X<0 X=64; R.
  470 IF X>64 X=0;Y=Y+16; IF Y>96 Y=0; R.
  480 IF Y>96 Y=0;X=X+8; IF X>64 X=0; R.
  490 RETURN
  500fFOR I=0 TO 8; C=1; REM STORE KARAKTER IN ROM-DATA
  510 FOR J=#9600+I TO #8900 STEP -#200
  520 C=C*2; IF ?J&1 C=C+1
  530 NEXT J
  540 IF I<8 B?(#80*I+A)=C
  550 IF I=8 B?(A&#F0*8+#10+A&#F)=C
  560 NEXT I
  570 RETURN
```

GRAFISCH PRINTEN met de Microline 80

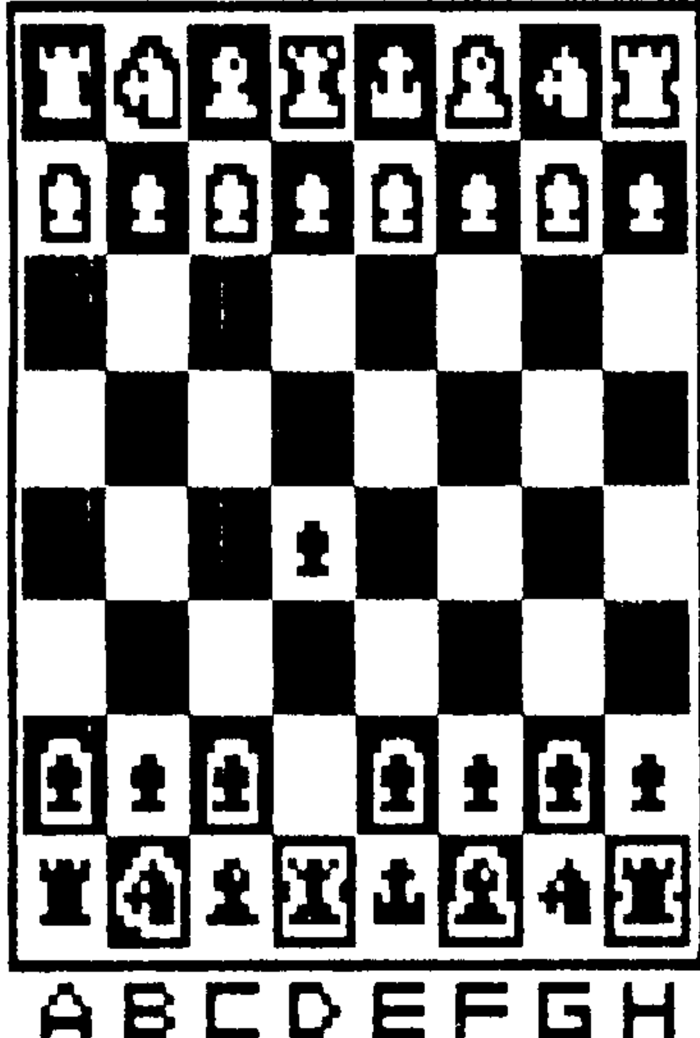
Door : Martin Janssen

In AN 1.8 stond reeds een programma om grafisch te printen in 'clear 4'. Het onderstaande programma print in mode 1, 2, 3 en 4, en onderscheidt zich verder in de volgende opzichten:

- alle grafische tekens worden vertaald in de hex codes 80-9F en E0-FF, zodat A0-DF vrij blijven voor KANA- of zelf te definieren symbolen.
- bij geinverteerd printen blijft de inhoud van het video geheugen onaangetast.
- na het printen wordt de printer teruggezet naar zijn standaard instelling.
- het programma is ongeveer half zo lang!

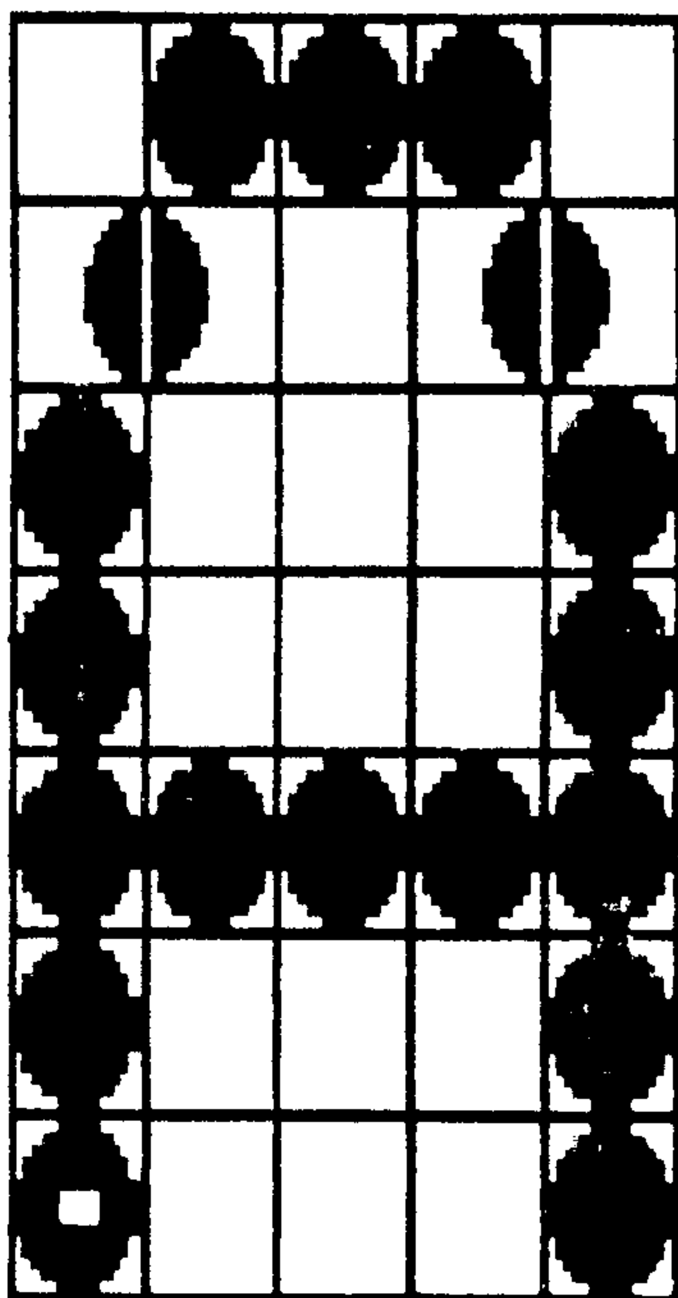
De gebruiksaanwijzing is ongeveer gelijk. Het is uiteraard ook geschreven volgens de standarisatienorm bit 3 van de 8255.
Voor de zekerheid nog even dit. Het onderstaand programma print zelf nog niks, maar genereert een machinecode-programma (assembler) dat gebruikt kan worden om grafisch te printen. Dit assemblerprogramma start op adres #7000 (eventueel regel 60 aanpassen).
Je kunt het printprogramma aanroepen met LINK#7000. Druk na de piep op een willekeurige toets om het printen te starten. Tik een 'I' in als je 'geinverteerd' wil printen. Nadat de printer is uitgerateld, moet evenals bij het programma van Leendert, nog een CTRL F worden gegeven omdat het beeldscherm is uitgeschakeld tijdens het printen.
Dit kan ook auto door regel 200 te vervangen door: P!4=#60EA0603; R.
Tenslotte wil ik er nog op wijzen dat volgens de fabrikant de 'duty cycle' tijdens grafisch printen maximaal 40% van de 'character printing duty cycle' mag zijn. Wat ze hiermee precies bedoelen is mij niet duidelijk. Wel lijkt het mij verstandig om te voorkomen dat de printpagina voor het grootste deel zwart is.

```
>L.
   10 REM GRAFISCH PRINTEN MET DE MICROLINE 80
   20 B=#B000;C=#60;D=#61;E=#62;F=#64;G=#65;H=#69;K=#FFF4
   30 DIM LL9;FOR I=0 TO 9;LLI=#8000;N.
   40 P.$21;GOS.z;GOS.z;P.$6'"EINDADRES="&P+8;END
   50
   60z P=#7000;[;
   70:LL0 JSR#FD1A;JSR#FE94;LDX@0;CMP@#49;BNE LL1;DEX;:LL1 STX H
   80 JSR#F7D1;];!P=#38181502;P!4=#EA0D1D;P=P+7;[;
   90 LDA B;ROL A;ROL A;ROL A;AND@3;TAX;LDA#FECF,X;STA D;LDA@#1F
  100 CPX@3;BEQ LL2;LSR A;:LL2 STA C;LDA@0;STA E;LDA@#80;STA E+1
  110:LL4 LDA@3;STA F;LDY@0
  120:LL6 LDA(E),Y;LDX@3
  130:LL7 ASL A;ROR G,X;ASL A;ROR G,X;DEX;BPL LL7
  140 TYA;SEC;ADC C;TAY;DEC F;BNE LL6;LDA@8;STA B+2;LDX@3
  150:LL8 LDA G,X;EOR H;CMP@#80;ROR A;SEC;ROR A;JSR K
  160 DEX;BPL LL8;LDA@0;STA B+2
  170 INC E;BNELL9;INC E+1;:LL9 LDA E;AND C;BNE LL4;LDA@#D;JSR K
  180 LDX C;INX;TXA;ASL A;ADC E;STA E;LDA E+1;ADC@0;STA E+1
  190 CMP D;BCC LL4;JSR#F7D1;];!P=#0D1E361B
  200 P!4=#60EAEA03; R.
```



VOORBEELD van GRAFISCH

PRINTEN in MODE 2.



Zó ziet het scherm eruit.
Bij het printen is het plaatje in de lengte-richting uitgerekt, waardoor de cirkels niet meer rond zijn.
Let op het witte vierkantje links-onder. Dit is de "cursor".

ATOM CHESS EN MJCOS

De meeste programma's kun je gemakkelijk converteren naar MJCOS formaat. Bij het schaakprogramma ATOM CHESS lukt dit niet, omdat dit programma is beveiligd tegen (illegaal) kopieren. Dit programma kun je niet eerst laden en daarna runnen, ook al ken je het beginadres (#378A). Door een aantal wijzigingen aan te brengen in de programma code lukt het toch om het programma te laden op 300 baud, en vervolgens te saven via MJCOS. Hiervoor ga je als volgt te werk:

1. Laadt het programma d.m.v. het gebruikelijke *RUN"CHESS". Druk na het laden op BRK!

2. Type in:

```
Q=#28FD
!Q=#0D29024C
Q!5=#6C20F1D0
Q!9=#0D9C9028
Q!16=#20202E4C
!#296E=#20202E53
!#3769=#4CEAEAEA
```

Hiermee herstel je o.a. de schade die is aangericht door de BRK.

3. LINK naar MJCOS (beginadres).

4. Schrijf het programma op de MJCOS manier weg door in te tikken:
   *S. "FAST-CHESS" 2800 3C00 378A, 9000 9800

Dit programma kun je in het vervolg laden met *L."FAST-CHESS" en daarna runnen met LINK#378A, of ineens met *R."FAST-CHESS". Ook kun je dit programma na een BRK herstarten via LINK#378A. Het saven en laden van de bordstand gaat ook via MJCOS. Niet vergeten om na een BRK opnieuw MJCOS te linken. Veel succes!

WORDPACK ZWART OP WIT

Met de volgende wijzigingen in de WORDPACK ROM verschijnt de tekst in zwarte letters op een wit scherm. Iedereen is het er over eens dat dit een prettiger en leesbaarder beeld geeft.

```
P=#AB8C;[;DEC#95;LDA@#FF;NOP;]
P=#AC4B;[;LDA@#FF;STA#E1;LDY@#80;STY#95;LDY@0;STY#94;]
FOR I=#AD00 TO #AFFF; ?I=?I:#FF; NEXT I
```

Martin Janssen

GROOTMEETER STRIJDEN



ZELF EEN SCHAAKPROGRAMMA MAKEN?

Veel programmeurs (zelfs ervaren) schrikken ervoor terug om zelf een schaakprogramma te maken.De reden is gewoonlijk altijd dezelfde.Men meent dat men zelf een zeer goed schaker moet zijn om een schaakprogramma te schrijven.Niets is minder waar. KEN THOMPSON de programmeur van BELLE,(de huidige wereldkampioen computerschaak) antwoordde op de vraag naar zijn eigen speelsterkte,"I don't play chess".
De enigste vereiste is dat men de spelregels moet kennen; dit geldt voor elk spel wat men wilt programmeren.De spelregels vindt men in elk schaakboek voor beginners en zijn niet zo ingewikkeld als zij op het eerste zicht lijken.Men kan zich afvragen waarom men zelf een schaakprogramma zou schrijven als er al reeds zoveel op de markt zijn. Hierop zijn twee antwoorden. 1)De op de markt zijnde schaakprograam's (voor de atom) spelen relatief slecht.2)Het zelf schrijven van een schaakprogramma is een uitdaging voor het intellect. Men leert niet alleen programmeren maar men werkt aan een probleem dat nooit opgelost zal zijn.Een programma dat in elke stelling DE beste zet zal kiezen bestaat NIET.Elk programma zal voor verbetering vatbaar blijken te zijn en het zoeken van deze verbeteringen vormen weer een wetenschap op zich zelf. Het is niet mijn bedoeling U reeds op voorhand de moed te ontnemen. Neen integendeel. Ik wil enkel maar aantonen dat het schrijven en verbeteren van een schaakprogramma een hobby kan zijn waar men altijd kan mee bezig zijn omdat men nooit zal kunnen zeggen dat het programma af is; de uitdaging zal altijd blijven bestaan.
Ik hoop dat ik U genoeg heb kunnen aansporen om de moed op te brengen er zelf aan te beginnen. Om u verder op weg te helpen wil ik nu enige antwoorden geven op de automatisch opkomende vragen.Welke taal te kiezen? Hoe weet de komputer waar een stuk staat op het bord? Hoe weet hij welk stuk het is enz...

HET BORD.

110	111	112	113	114	115	116	117	118	119
100	101	102	103	104	105	106	107	108	109
90									99
80									89
70									79
60									69
50									59
40	A3								49
30									39
20	A1		C1					H1	29
10	11	12	13	14	15	16	17	18	19
0	1	2	3	4	5	6	7	8	9

Het schaakbord bestaat uit 64 velden.De meest voor de handliggende methode is dus een array van 64 velden. Hiermede is het probleem echter niet opgelost. Het schaakbord is tweedimensionaal en onze array ééndimensionaal; de velden liggen als het ware op één rij achter elkaar. Veld A2 ligt dus onmiddelijk achter veld H1. Men dient dus rekening te houden met de rand. Hiervoor bestaan verschillende methoden.Wij zullen ons echter beperken tot één;dit voor de duidelijkheid.De gemakkelijkste methode is het

bijvoegen van een extra randveld.Dit volstaat niet daar het paard door zijn mogelijkheid om over stukken heen te springen ook over het randveld kan springen. We moeten dus twee randvelden voorzien en dan is dit probleem opgelost. We hebben dus een array nodig van 12 X 12=144 geheugenplaatsen. Daar onze array ééndimensionaal is volstaan we met 120 velden daar de velden 29,30 of 39,40 tweemaal voorkomen in een tweedimensionale array.

HET BEREKENEN DER ZETTEN.

Het berekenen der zetten is nu niet zo ingewikkeld meer. Laten we even het bord bekijken. Eén stap naar boven komt overeen met de waarde 10. Eén stap links vooruit met de waarde 9 en één stap rechts vooruit met de waarde 11. Het is nu niet moeilijk om de waarden voor de andere zetten de berekenen. Nog een voorbeeld. Het paard heeft wanneer het op het midden van het bord staat 8 mogelijke velden waar het naartoe kan.De waarden van de paardensprong komen overeen met de volgende getallen: +12,-12,+8,-8,+19,-19,+21,-21.
Al deze waarden worden in een tabel geplaatst. Door de waarde van het stuk op te tellen bij het adres van oorsprong bekomt men het adres van aankomst.

DE STUKKEN.

Met de adressen alleen zijn we natuurlijk nog niets.Op deze adressen moeten nog de waarden komen van de stukken. Welke waarde wij aan een bepaald stuk willen geven hangt volledig van onszelf aan.De manier die ik nu aangeef is slechts een voorbeeld.
pion = 1; paard =2; loper = 3; toren =4;dame =5; koning =6.
Daar we op elke geheugenplaats één byte kunnen opslaan, en we slechts 3 bits nodig hebben om de waarde van het stuk aan te geven, houden we nog 5 bits over om bijkomende informatie op te slaan; zoals de kleur van het stuk; Rokade is nog toegestaan; stuk heeft bewogen etc...
Samengevat :
bit 0-2: bevat het type van het stuk; pion,paard etc...
bit 3: 0= stuk heeft niet gespeeld. 1= stuk heeft gespeeld
bit 4: (enkel voor de koning) 0= rokade toegestaan.1= niet toegestaan
bit 5-6: vrij voor eigen informatie.
bit 7:0= wit stuk. 1= zwart stuk.
Lege velden worden op 0 gezet en de randvelden op #FF.
Een programma dat dus de zetten berekent kan er alzo uitzien.
Zoek het bord af naar waarden die niet gelijk zijn aan 0 of # FF (lege velden en randvelden). Heeft men een stuk gevonden test dan bit 7 om te weten of het een stuk is van de speler aan zet.Indien niet dan zoekt men verder tot het einde van het bord. Is het stuk van de juiste kleur kijk dan om welk type van stuk het gaat. Zoek in de bijbehorende tabel

de waarde op die men bij moet tellen om het adres van het veld van aankomst te berekenen. Kijk vervolgens welke waarde op het veld van aankomst staat. Is het een leeg veld? een randveld? of is het veld bezet door een stuk? Is het een eigen stuk of een vijandig stuk? Alle mogelijke (legale) zetten worden in een lijst geplaatst. Waarna men hieruit een zet moet kiezen en dit is nu het grootste probleem bij computerschaak (ook bij het gewone schaak).

DE TAAL:

Alvorens hierop in te gaan zullen we eerst een kleine berekening maken. Het gemiddelde aantal mogelijke zetten in een gemiddelde schaakstelling bedraagt ongeveer 35. Dit wil zeggen: wit is aan zet en kan kiezen uit 35 mogelijkheden; zwart kan hierop op 35 verschillende manieren antwoorden. Laat ons even doorrekenen.

1) 1°zet van wit = 35^1 = 35 mogelijkheden
2) 1°zet van zwart = 35^2 = 1225 totaal aantal mogelijkheden
3) 2°zet van wit = 35^3 = 42875 " " "
4) 2°zet van zwart= 35^4 = 1500625 " "
5) 3°zet van wit = 35^5 = 52521875 " "

U ziet een enorm aantal mogelijkheden. Als men rekening houdt met de toegestane bedenktijd in officiële wedstrijden van 40 zetten in 2 uur of een gemiddelde bedenktijd van 3 minuten per zet dan betekent dit nog dat we per seconde ± 290000 mogelijkheden moeten onderzoeken. Bedenk hierbij dat we dan nog maar 3 zetten ver gerekend hebben.
Natuurlijk hoeven we niet alle 52521875 mogelijkheden te berekenen. Dit voorbeeld dient alleen maar om aan te tonen hoe enorm snel er dient gerekend te worden en naar mijn bescheiden mening is een basic programma hiervoor te traag. Ik moet dus iedereen die een schaakprogramma wilt schrijven aanraden dit in assembler te doen. Clubleden die belangstelling hiervoor hebben kunnen mij altijd kontakteren.

Vansteenkiste R.

PROJECT:80-KOLOMS VIDEO DISPLAY UNIT.

Iedereen die de Elektuur van September heeft gelezen zal daarin de beschrijving van een 80-koloms VDU gezien hebben. Dit ontwerp komt in zeer veel opzichten overeen met het door mij op papier gezette idee. Zo veel zelfs dat het mij beter lijkt het helemaal over te nemen zodat we een hoop ontwikkeltijd besparen. Het enige wat we nog zouden kunen toevoegen is de geplande omschakeling naar 40- of 64-karakters en de aanpassing van de characterset indien dit nodig mocht blijken.
De Elektuur kaart kan natuurlijk niet rechtstreeks op onze bus aangesloten worden. We zouden hier een verloop plug moeten toepassen, dit lijkt mij echter geen onoverkomelijke moeilijkheid.

Op de print moeten nog enkele draadbruggen gelegd worden in verband met de adresdekodering. Iedereen kan de kaart dus plaatsen waar hem dat het beste uitkomt.
Mijn voorstel hiervoor luidt als volgt:

De CRTC op: #BFF0 en #BFF1
Het beeldschermgeheugen op: #9800 t/m #9FFF.

Hiertoe moeten de volgende verbindingen gelegd worden:

Ingang N37	aan Adreslijn.	Ingang N38	aan Adreslijn.
A	A15	F	A15
B	NA14	G	NA14
C	NA13	H	A13
D	A12	I	A12
E	A11	J	A11
		K	A10
		L	A9
		M	A8
		N	A7
		O	A6
		P	A5
		Q	A4
		R	NA3

De CRTC neemt nu door het niet volledig uitdekoderen van het adres (A2 en A1) 8 bytes in beslag. Dit lijkt mij echter wel acceptabel.
NB:Waar in de tabel een "N" voor de adreslijn staat wordt de geinverteerde waarde bedoeld.

De benodigde driver en initialisatie routine's zal ik volgende keer behandelen daar ikzelf de kaart nog niet in mijn bezit heb en hem dus zeker nog niet kan testen.

Will Verhoeven

Op sommige computers, zoals de BBC en Commodore machines, is het mogelijk om, terwijl de computer bezig is, alvast vooruit te typen. U hoeft dan niet te wachten met het intypen van opdrachten en andere invoer totdat de computer weer naar het toetsenbord kijkt.
Onderstaand programma verzorgt deze "type-ahead" faciliteit op de Acorn Atom. Met timer 1 van de VIA (op het board moet link LK2 gesloten zijn) wordt regelmatig een interrupt opgewekt, waarbij wordt gekeken of een toets is ingedrukt. Zo ja, dan wordt de ASCII-waarde opgeslagen in een buffer. Als er weer invoer nodig is, wordt door een nieuwe OSRDCH routine die ASCII-waarde weer uit de buffer gehaald.
Is de buffer leeg, dan wordt doorgesprongen naar de vorige OSRDCH routine die werd bewaard bij het opstarten.
Verandert na het opstarten de OSRDCH vector (bijv. doordat U GRMOD inschakelt), dan wordt automatisch doorgesprongen naar die nieuwe routine. U hebt daar dus geen omkijken meer naar! (De IRQ-service routine controleert steeds of de vector is veranderd, en handelt dienovereenkomstig).
Omdat de echo van de vooruit ingetypte karakters ontbreekt, worden een paar toetsen uitgeschakeld (alleen bij vooruit typen). Dit zijn de REPT-toets, omdat je niet ziet hoeveel karakters zijn opgewekt, en de copy- en cursor-toetsen, omdat je niet blind kunt editen. Door de speciale afhandeling van de lock-toets wordt deze ook genegeerd. Verder wordt gewacht totdat de ESC-toets wordt losgelaten voordat er karakters uit de buffer worden gelezen. Dit om vooruit ingetypte commando's niet in de kiem te smoren doordat ESC nog is ingedrukt.
Als page mode aanstaat en U reageert na een vol scherm met een "normale" toets, dan wordt die ook in de buffer opgeslagen. Als U dat niet wilt dan met een van de genegeerde toetsen (COPY, LOCK of CURSOR) reageren.
Bij COS-operaties kun je niet vooruit typen, omdat dan de Interrupt uitgeschakeld wordt.
Als de buffer vol is, wordt dit aangegeven met een piepje. Verder doortypen heeft geen zin, dus eerst wachten tot de computer weer invoer nodig heeft en er dus ruimte in de buffer ontstaat.
De lengte van de buffer kunt U instellen in regel 30 (max.255). In regel 950 kunt U het beginadres van de buffer veranderen. De buffer staat nu direct achter de object kode. In regel 40 kunt U eventueel een ander werkgebied kiezen (bijv.zero-page #B0 e.v.). Wilt U het werkgebied direct achter de buffer dan een regel toevoegen:

```
   955 F=B+L
```

Het werkgebied, dat niet door andere programma's benut mag worden, wordt als volgt gebruikt:

```
 F       :aantal karakters in de buffer.
 F+1     :tijdelijke opslag X en Y registers
 F+2     :index naar eerst te lezen karakter in de buffer.
 F+3     :index naar eerste vrije positie in de buffer.
 F+4     :toetscode laatste toets.
 F+5,F+6:vector naar oude OSRDCH routine.
 F+7,F+8:indirect jumpadres bij bepalen ASCII-code.
```

Ronald Boers

Listing:

```
 10 REM /// TYPE AHEAD /// V1.2
 20 IN."BASISADRES (BV.#6000)"Q;B=Q
 30 L=100;REM LENGTE BUFFER
 40 F=#21C;REM F T/M F+8 WORDEN GEBRUIKT
 50 DIM II9,RR4,TT4;FOR Q=0TO19;IIQ=0;N.Q
 60 P.$21;FOR I=1TO2;P=Q;[
 70\*************************
 80\initialisation
 90\NEW IRQ-SR.,OSCLI,OSRDCH
100 JSR II6
110 LDX@3
120:TT0 LDA TT1,X;STA#204,X
130 DEX;BPL TT0
140\INIT VIA TIMER 1
150 LDA@#40;STA#B80B;LDA@#C0;STA#B80E
160 LDA@#FF;STA#B804;STA#B805;STA#B807
170\CLEAR WORK AREA
180 LDA@0;LDX@4
190:TT3 STA F,X;DEX;BPL TT3
200 CLI;RTS
210:TT1;];!P=II0;P!2=TT2;P=P+4;[
220\*************************
230\new oscli
240\AVOID IRQ DURING *CAT
250:TT2 SEI
260 JSR#F8EF NORMAL OSCLI
270 CLI;RTS
280\*************************
290\new osrdch
300:RR0 PHP;STX F+1
310\CALLED BY OSFIND (#FC38)?
320\HIGH BYTE CALLER = #FC
330 TSX;LDA#103,X;LDX F+1
340 CMP@#FC;BEQ RR1
350\ELSE ENABLE IRQ
360 PLP;CLI;PHP
370\BUFFER EMPTY?
380:RR1 LDA F;BNE RR2
390\CALL PREVIOUS OSRDCH
400 SEI;JSR RR4
410\GET KEY NUMBER
420 PHA;TXA;PHA;TYA;PHA
430 JSR#FE71;STY F+4
440 PLA;TAY;PLA;TAX;PLA;PLP;RTS
450\WAIT UNTIL ESC RELEASED
460:RR2 LDA#B001;AND@#20;BEQ RR2
470\GET ASCII-CODE FROM BUFFER
480 STY F+1;LDY F+2;LDA B,Y
490\WRAPAROUND IN BUFFER NEEDED?
500 INY;CPY@L;BCC RR3;LDY@0
510:RR3 STY F+2;DEC F
520 LDY F+1;PLP;RTS
530:RR4 JMP(F+5)
```



```
540\***************************
550\irq-service routine
560:II0 LDA#B804 RESTART TIMER 1
570 TXA;PHA;TYA;PHA
580\PROTECT AGAINST GRAPHICS SWITCHING
590 LDA@0;JSR#FE85
600\STILL SAME OSRDCH?
610 LDA#20A;CMP@RR0%256;BNE II7
620 LDA#20B;CMP@RR0/256;BEQ II8
630:II7 JSR II6
640\SCAN KEYBOARD
650:II8 JSR#FE71;BCS II4
660\SKIP SOME SPECIAL KEYS:
670\LOCK,COPY,ESC,CURSOR KEYS
680 TYA;LDX@4
690:II9 CMP TT4,X;BEQ II4
700 DEX;BPL II9
710 CPY P+4;BEQ II2;STY P+4
720 JSR II3
730\BLEEP IF BUFFER FULL
740 LDY P;CPY@L;BCS II5
750\STORE ASCII-CODE IN BUFFER
760 LDY P+3;STA B,Y;INY
770\WRAPAROUND IN BUFFER NEEDED?
780 CPY@L;BCC II1;LDY@0
790:II1 STY P+3;INC P
800:II2 PLA;TAY;PLA;TAX;PLA;RTI
810\GET ASCII-CODE
820:II3 PHP;LDX@#17;JSR#FEC5
830 LDA#FEE3,X;STA P+7;LDA@#FD
840 STA P+8;TYA;JMP(P+7)
850:II4 LDA@#FF;STA P+4;BNE II2
860:II5 JSR#FD1A;BMI II2
870\SAVE OLD OSRDCH
880:II6 LDA#20A;STA P+5
890 LDA#20B;STA P+6
900\NEW OSRDCH
910 LDA@RR0%256;STA#20A
920 LDA@RR0/256;STA#20B;RTS
930:TT4;];!P=#0506070E;P?4=#3B;P=P+5
940 REM***********************
950 B=P;REM BUFFERADRES
960 N.I;P.$6
970 @=0;P."CODE VAN #"&O" TOT #"&P'
980 P."BUFFER VAN #"&B" TOT #"&B+L'
990 P."OPSTARTEN MET:LINK#"&O''
1000 END
   0 (C) RONALD BOERS 6-8-1983
```



UITBREIDING TOETSENBORD.

De standaard Atom beschikt alleen over een (standaard) toetsenbord voor de invoer van opdrachten. Om verschillende redenen kan het handig zijn over andere manuele invoer faciliteiten te beschikken. Daarbij valt te denken aan joysticks, numerieke en meer uitgebreide toetsenborden. Voor al dit soort uitbreidingen kunnen aanpassingen worden gemaakt. Het zou echter handig zijn als we een hardwarematige aanpassing zouden maken die zo min mogelijk software veranderingen nodig heeft.

Het toetsenbord wordt uitgelezen met behulp van drie poorten; A,B en C van de PPIA (ic 25). Deze zijn geadresseerd op respectievelijk £B000, £B001 en £B002 (zie ATP). Daartoe is het toetsenbord verdeeld over een matrix zoals te zien is in het electrisch schema dat bij de Atom wordt verstrekt. De laagste vier bits van uitvoer poort A maken het mogelijk de lijnen 0 t/m 9 van ic 26 beurtelings laag te trekken, zodat een rij uit de matrix onderscheiden kan worden. Met bit 0 t/m 5 van invoer poort B kunnen de kolommen van de matrix worden gelezen. Bit 6 en 7 van poort B kunnen respectievelijk de CTRL en de SHIFT toets lezen. De REPT toets is te lezen uit bit 6 van poort C.

Nu zou men zich kunnen voorstellen de betrokken poorten naar buiten te halen, maar dan zou voor ieder toetsenbord of joystick een extra ic 7445 nodig zijn. Daarom willen wij voorstellen: Breng de matrix naar buiten en de lijnen naar bit 6 en 7 van poort B en bit 6 van poort C. In fig 1 geven wij aan hoe wij ons dat voorstellen. Aan de Atomzijde hebben wij een male-connector (pennen) gebruikt.

Suggesties voor inbouw:

De connector aan de achterkant van de Atom monteren (wij hebben daarvoor de koelplaat vervangen door een groter plaatje aluminium met daarin o.a. deze connector.) De draden worden op de soldeerzijde van de toetsen aangesloten.

De connector in de (linker) zijkant van de bovenkast (waar veel mensen de joystick hebben) en de aansluitingen op de soldeerzijde van de voetjes van ic 25 en 26.

Het gebruik:

Wil men nu een joystick of toetsenbord aansluiten dan zoekt men in de matrix op welke lijnen men nodig heeft. Bijvoorbeeld voor de letter A heeft men nodig: rij 6 en kolom 3 , op de connenctor pen 8 en pen 17. Verder kan van de reeds aanwezige software gebruik gemaakt worden. (Men hoeft geen programma's meer te veranderen; sluit de joystick op de juiste letter aan.)

D. Urbanik & L. Veltmeijer **(Voor fig.1 z.o.z.)**

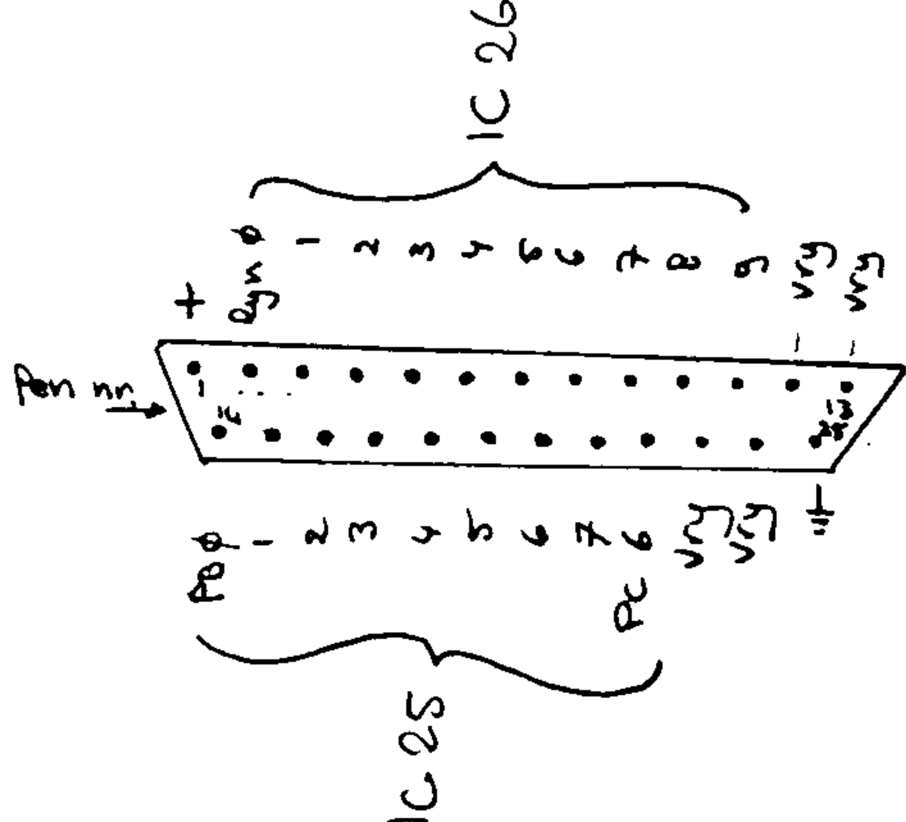
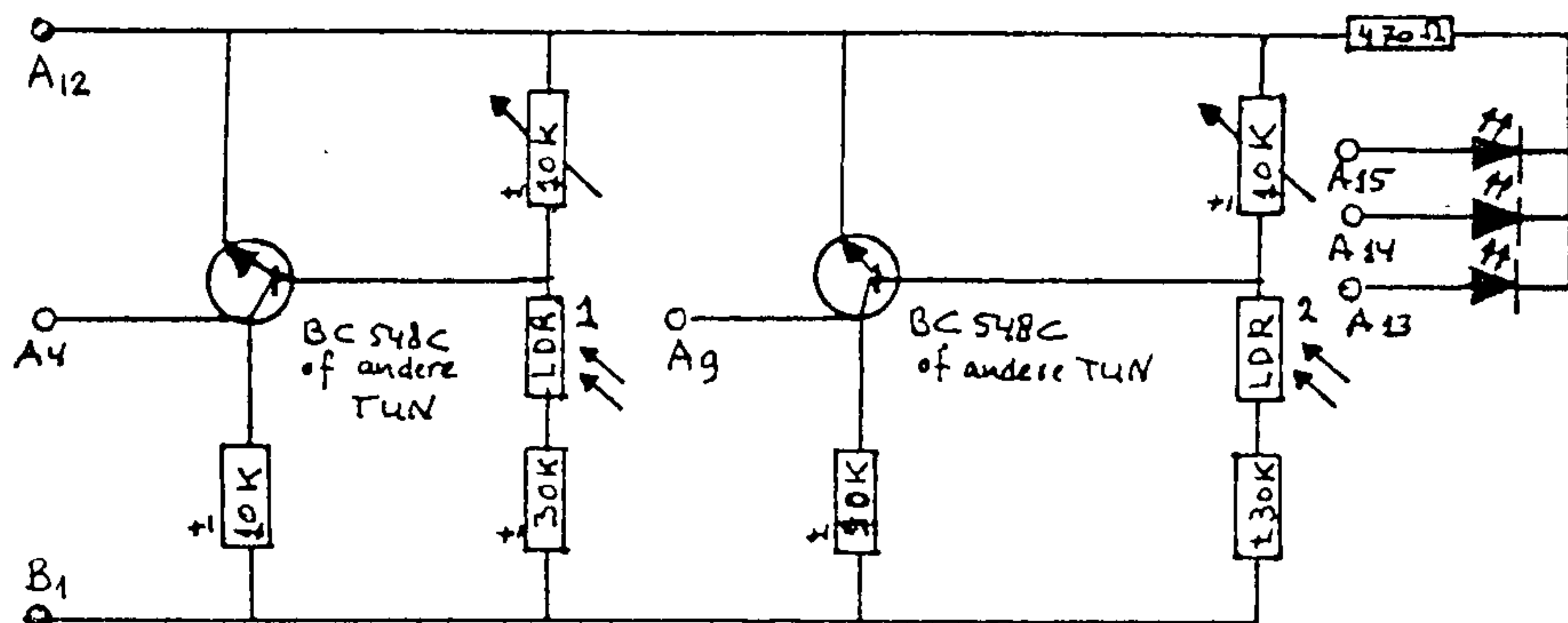


Fig. 1 Als plug : 25-polige D-connector
Nummering op de plug aangegeven.

```
)L.
   10 REM HOEDJE
   20 REM GERARD AKKERMANS
   30 REM FLOATING POINT & VOLLEDIG GRAPHIC-RAM NOODZAKELIJK
   40 REM TOTALE TEKENTIJD: 50 MINUTEN!
   50 REM VOOR ONGEDULDIGEN: AAN REGEL 120
   60 REM TOEVOEGEN "STEP 2" OF "STEP 3"
   70 REM INDIEN GEEN JOSBOX, DAN "80 CLEAR 4"
   80 GRMOD
   90 P=127;Q=99;Z=48
  100 %G=88.0;%V=44.0;%R=0.33;%U=3.0
  110 %P=%G*%G;%W=%P/FLT(Z*Z);%F=1.5*PI/%G
  120 FOR A=-Z TO Z
  130  %D=FLT(A*A)*%W;B=%(.5+SQR(%P-%D))
  140  M=A+P;N=Q-A
  150  FOR C=-B TO B
  160   %T=SQR(FLT(C*C)+%D)*%F
  170   S=%((SIN(%T)+%R*SIN(%U*%T))*%V)
  180   K=C+M;L=S+N
  190   PLOT 13,K,L;MOVE K,(L-1);PLOT 7,K,0
  200   NEXT C
  210  NEXT A
  220 DO; UNTIL 0
  230 END
```

TIJDMETING

Hallo, computergenoten! Weer een nieuw onderwerp. Niets sensationeels, maar wel leuk. Tijdmeting.
Het enige dat men moet hebben is de VIA en wat kleine electronische spullen
Hier volgt het schema.



Waarom niet alles aan de A- of B-poort? Gemakzucht! Het is gemakkelijker om te zeggen: alle B-lijnen zijn out en alle A-lijnen zijn in (of andersom). De LED's dienen voor foutendetectie en als indicatie van waar men mee bezig is.

Het programma werkt als volgt:
Men tikt RUN en als alles goed gaat brandt de groene LED. Bij verduistering van LDR 1 springt de oranje LED aan en het programma begint te tellen. Bij verduistering van LDR 2 springt de rode LED aan en stopt het tellen.
Voor het afstellen van de LDR's gebruikt men regel 550, want deze geeft de inhoud van ≠ B800 weer. Men moet wel eerst even de lijnen op input schakelen door ?≠B802 = 0. LDR 1 geeft dan 0 of 64 en LDR 2 0 of 2. Ze moeten beiden juist 0 aanwijzen en bij verduistering resp. 64 en 2.
Voor het fijn afregelen kan men in serie met de potmeters een instelpot van 100 ohm hangen.

Het programma is erop gebaseerd, dat het steeds telt en zo de plaatsen ≠80 t/m ≠ 83 gebruikt. Dan wordt het aantal klokpulsen geteld (regels 490/520).

Men kan het programma natuurlijk uitbreiden door een aantal metingen te doen en dan het gemiddelde te berekenen.

Wat voor metingen? Bijv. als uw zoon of dochter op school geplaagd wordt met de slingerproef, kunt u de stof met behulp van dit programma illustreren.
Geen goede vrienden met de buurman? Meet eens de snelheid waarmee hij naar zijn werk gaat en die waarmee hij thuis komt ...
Er is nog veel meer mee te doen. U ziet maar.

Jeroen Baten.

P.S. de timing is nog niet geijkt. Maar theoretisch klopt het!

```
10 REM Tijdmeting
15 P.$12
20 £=0
30 !≠80=00000000
40 ?≠B802=0
50 ?≠B803=≠FF
60 DIM VV2
70 FOR F=1 TO 2
80 P.$21
90 DIM P-1
100[
110:VV0 LDA£≠80
120 STA ≠B80F
130 LDA ≠B800
140 AND £≠42
150 CMP £64
160 BNE VV0
170 LDA £≠40;STA ≠B80F
180 LDX £≠0
190:VV1 INX
200 STX ≠80
210 LDA ≠B800
220 AND £≠42
230 CMP £2
240 BEQ VV2
250 CPX £≠FF
260 BNE VV1
270 LDX ≠81
280 INX
290 STX ≠81
300 CPX £≠FF
310 BNE VV1
320 LDX ≠82
330 INX
340 STX ≠82
350 CPX £≠FF
360 BNE VV1
370 LDX ≠83
380 INX
390 STX ≠83
400 CPX £≠FF
410 BNE VV1
420:VV2 LDA £≠20
430 STA ≠B80F
440 RTS
450 ]
460 N.
470 P.$6
480 LINK VV0
490 A=((?≠83*255*255*255)+(?≠82*255*255)+(?≠81*255)
    +?≠80)*21 (let wel: niet *12)
500 B=((?≠82*255*255)+(?≠81*255)+?≠80)*12
510 C=((?≠81*255)+?≠80)*12
520 D=(?≠80*12)
530 FP.(A+B+C+D)*1E-6 "SECONDEN "'
540 E.
550 £=0;DO P.?≠B800&≠42;U.0
```

£ = @

GELEZEN:Graphics — bron:HCC nieuwsbrief no.7-1983

```
 10 REM SPIRAAL
 20 CLEAR 4
 30 P=128;Q=91
 40 %S=0.1;%N=0
 50 MOVE P,Q
 60 %T=0;DO
 70 X=%(8*%T*SIN(%T+%N)+P)
 80 Y=%(8*%T*COS(%T+%N)+Q)
 85 IF X<0 OR X>255 OR Y<0 OR Y>191 THEN %N=%N+PI/4;GOTO 120
 90 DRAW X,Y
 95 %T=%T+%S
100 FUNTIL %T>5*PI
110 END
120 IF %N<2*PI THEN %T=0;MOVE P,Q ;G.100
130 G.90
```

MAAK VAN DE ACORN EEN FREQUENTIETELLER

OPMERKING VOORAF: HET VERDIENT AANBEVELING EEN DATASHEET AAN TE SCHAFFEN VAN DE 6522, OPDAT HET VERHAAL GEMAKKELIJKER TE VOLGEN IS

DE ACORN IS MET EEN KLEINE UITBREIDING OM TE "TOVEREN" IN EEN FREQUENTIETELLER. DOORDAT DE ACORN EEN DIGITAAL SYSTEEM IS, IS DEZE ALLEEN GESCHIKT OM BLOKGOLVEN TE VERWERKEN. INDIEN WE DUS EVENTUEEL EEN SINUS WILLEN VERWERKEN, ZULLEN WE DEZE EERST OM MOETEN ZETTEN IN EEN BLOKGOLF. DIT IS EVENTUEEL TE REALISEREN MET BV. EEN 555 TIMER. OP DIT ZAL IK ECHTER NIET VERDER INGAAN.

INDIEN WE DE FREQUENTIE VAN EEN PULSTREIN WILLEN METEN, DAN KUNNEN WE STELLEN: f=N/T WAARBIJ f=FREQUENTIE
N=AANTAL PULSEN
T=VERSTREKEN TIJD

HIERUIT KUNNEN WE OPMAKEN DAT INDIEN WE 1 SEC TELLEN, f=N
IN WOORDEN: DE FREQUENTIE VAN EEN PULSTREIN IS GELIJK AAN HET AANTAL PULSEN WELKE IN EEN SECONDE VERSTRIJKEN. (NIET BIJ ZEER LAGE FREQUENTIES)

OM DE FREQUENTIES TE TELLEN, MAKEN WE GEBRUIK VAN DE 6522.
DE T1 COUNTER WORD GEBRUIKT OM EEN PERIODE VAN 1 SECONDE TE METEN
DE T2 COUNTER WORD GEBRUIKT OM HET AANTAL PULSEN TE TELLEN.

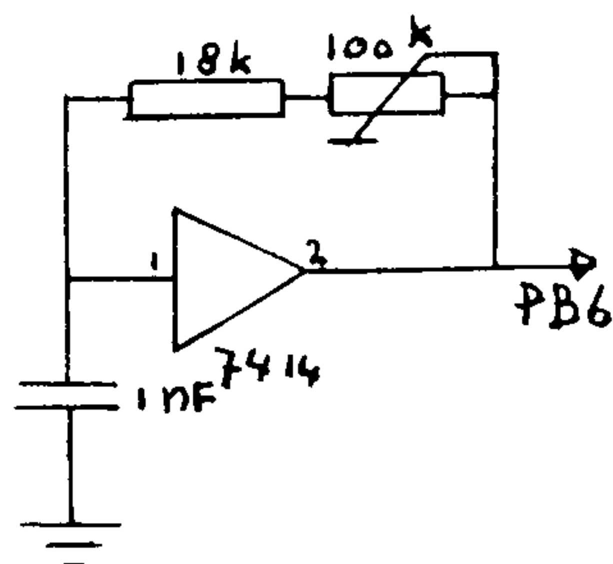
DE REGISTERS VAN DE 6522

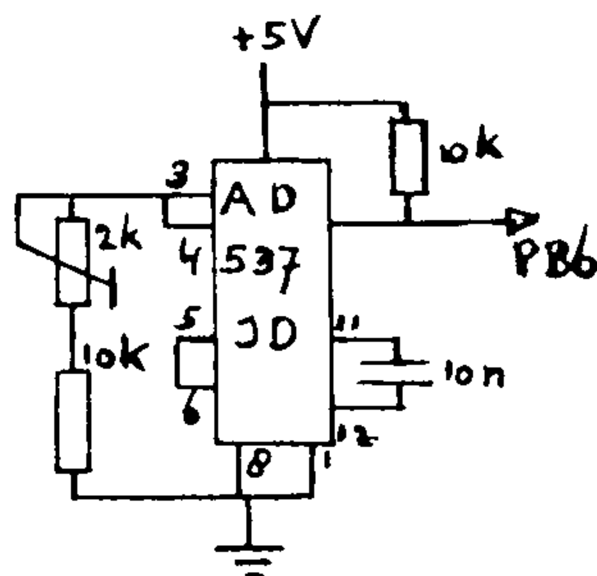
#B804=T1CL	TIMER 1 COUNTER LOW
#B805=T1CH	TIMER 1 COUNTER HIGH
#B806=T1LL	TIMER 1 LATCH LOW
#B807=T1LH	TIMER 1 LATCH HIGH
#B808=T2CL	TIMER 2 COUNTER LOW
#B809=T2CH	TIMER 2 COUNTER HIGH
#B80B=ACR	AUXILIARY CONTROL REGISTER
#B80C=PCR	PERIPHERAL CONTROL REGISTER
#B80D=IFR	INTERRUPT FLAG REGISTER
#B80E=IER	INTERRUPT ENABLE REGISTER

HET HIERBIJ GEGEVEN PROGRAMMA IS NOG LANG NIET OPTIMAAL, EN LEENT ZICH DAN OOK UITSTEKEND VOOR EIGEN EXPERIMENTEN
EEN VAN DE BEPERKINGEN VAN DIT PROGRAMMA IS, DAT DE FREQUENTIE ZICHT NIET BOVEN DE 65KHz MAG BEGEVEN, EN BENEDEN DE 10 Hz ONNAUWKEURIGE RESULTATEN GEEFT. DIT IS MET VERANDERING VAN DE SOFTWARE WEL OP TE VANGEN.
HIERONDER NOG EEN TESTSCHAKELING, EN EEN APPLICATIE
DE GEGEVEN TESTSCHAKELING GEEFT EEN BLOKGOLF MET EEN FREQEUNTIE VARIEREND TUSSEN DE 4400 EN 30000 Hz
DE APPLICATIE IS EEN MOGELIJKHEID OM MET DE ACORN EEN TEMPERATUURMETER TE MAKEN. EEN VAN DE MOGELIJKHEDEN VAN EEN TEMPERATUURMETER IS BV. EEN CV STURING TE MAKEN

TESTSCHAKELING

APPLICATIE





PROGRAMMA

```
 10 REM FREQUENTIETELLER
 20 DIM BB0
 30 BB0=0
 40 P.$21
 50 FOR N=1 TO 2;P=#2800
 60[
 70 LDA#B804
 80 INC#80
 90 PLA;RTI
100]
110 N.
120 FOR N=1 TO 2;P=#2820
130[
140 LDA@0;STA#80
150 STA#B808;STA#B809
160 LDA@#60;STA#B80B
170 LDA@#C0;STA#B80E
180 LDA@#50;STA#B804
190 LDA@#C3;STA#B807
200 STA#B805
210 LDA@0;STA#204
220 LDA@#28;STA#205
230 CLI
240:BB0
250 LDA#80
260 CMP@20;BNEBB0
270 LDA#B808;STA#81
280 LDA#B809;STA#82
290 LDA@0;STA#83;STA#84
300 RTS
310]
320 N.;P.$6$12
330 LINK#2820;A=#FFFF-!#81
340 IF A=#FFFF THEN A=0
350 P.A;P.$13;G.330
```

BESCHRIJVING PROGRAMMA

REGEL 70: BEGIN INTERRUPT ROUTINE. MET DEZE INSTRUKTIE WORD HET INTERRUPTBIT IN HET IFR GERESET

REGEL 80: TEL HET AANTAL INTERRUPTS

REGEL 90: HAAL ACCU VAN STACK, EN KEER TERUG NAAR HET VORIGE PROGRAMMA

REGEL 140: ZET HET AANTAL INTERRUPTS OP 0

REGEL 150: ZET TIMER 2 OP 0

REGEL 160: ZET TIMER 1 OP FREE RUN MODE, EN ZET TIMER 2 OP PULSE COUNTING MODE
FREE RUN MODE HOUDT IN, DAT T1 LEEG TELT OP DE KLOK-FREQUENTIE, EN OP HET MOMENT DAT 0 BEREIKT WORD, DE TIMER GELADEN WORDT MET DE INHOUD VAN DE LATCH OP HET MOMENT VAN 0 WORDT DAARBIJ EEN INTERRUPT NAAR DE COMPUTER GEGEVEN.

REGEL 170: ENABLE INTERRUPT VAN T1

REGEL 180: LAAD T1 EN DE LATCH VAN T1 MET DECIMAAL 50000
20 INTERRUPTS VAN 50000(=50 mS) MAKEN SAMEN EEN PERIODE VAN 1 SEC

REGEL 190: IDEM

REGEL 200: IDEM

REGEL 210: ZET INTERRUPTVECTOR OP #2800
REGEL 220: IDEM
REGEL 230: ENABLE INTERRUPTBIT VAN DE PROCESSOR
REGEL 250: CONTROLEER OF DE PERIODE VAN 1 SEC AL VERSTREKEN *is.*
REGEL 260: IDEM
REGEL 270: VERPLAATS DE INHOUD VAN T2 NAAR GEHEUGENLOKATIE #81 TOT EN MET #84, ZODAT HET VIA BASIC IS UIT TE LEZEN
REGEL 280: IDEM
REGEL 290: IDEM
REGEL 300: KEER TERUG NAAR BASIC
REGEL 330: MEET DE FREQUENTIE, EN BEREKEN DAARNA DE JUISTE WAARDE. DIT IS NODIG OMDAT OOK T2 NAAR BENEDEN TELT, DIT HOUDT IN DAT INDIEN 1 PULS GETELT IS, DE INHOUD VAN T2 #FFFF IS.
REGEL 350: PRINT DE FREQUENTIE, EN BEGIN OPNIEUW.

Fred Appelman

```
LIST
   10 REM
   20 REM DRIEHOEK VAN
   30 REM    PASCAL
   40 REM
   50 @=0;N=0
   60 C=42 ; REM LENGTE ARRAY
   70 DIM A(C),B(C)
   80 REM ARRAYS SCHOONMAKEN
   90 FOR X=0 TO C
  100 A?X=0 ; B?X=0
  110 NEXT X
  120 B?(C/2)=1 ; REM TOP VAN
  130 REM             DRIEHOEK
  140 GOSUB 340
  150 FOR Z=0 TO 10
  160 GOSUB 210 ; REM TEL OP
  170 GOSUB 340 ; REM PRINT B-ARRAY
  180 NEXT Z
  190 END
  200
  210 REM OPTEL ROUTINE
  220 X=2
  230 IF N%2=1 X=X+1
  240 DO
  250 A?X=B?(X-1)+B?(X+1)
  260 X=X+2
  270 UNTIL X>(C-2)
  280 REM A NAAR B , WIS A
  290 FOR X=0 TO C
  300 B?X=A?X ; A?X=0
  310 NEXT X
  320 N=N+1
  330 RETURN
  340 REM PRINT B-ARRAY
  350 FOR X=0 TO C
  360 IF B?(X)=0 PRINT"   ";GOTO 400
  370 IF (B?(X)/100)<1 PRINT " "
  380 IF (B?(X)/10)<1 PRINT " "
  390 PRINT B?X
  400 NEXT X
  410 PRINT '
  420 RETURN
```

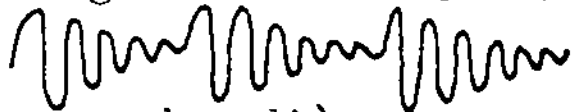
```
                                        1
                                     1     1
                                  1     2     1
                               1     3     3     1
                            1     4     6     4     1
                         1     5    10    10     5     1
                      1     6    15    20    15     6     1
                   1     7    21    35    35    21     7     1
                1     8    28    56    70    56    28     8     1
             1     9    36    84   126   126    84    36     9     1
          1    10    45   120   210   252   210   120    45    10     1
       1    11    55   165    74   206   206    74   165    55    11  .  1
```

TOONKUNST

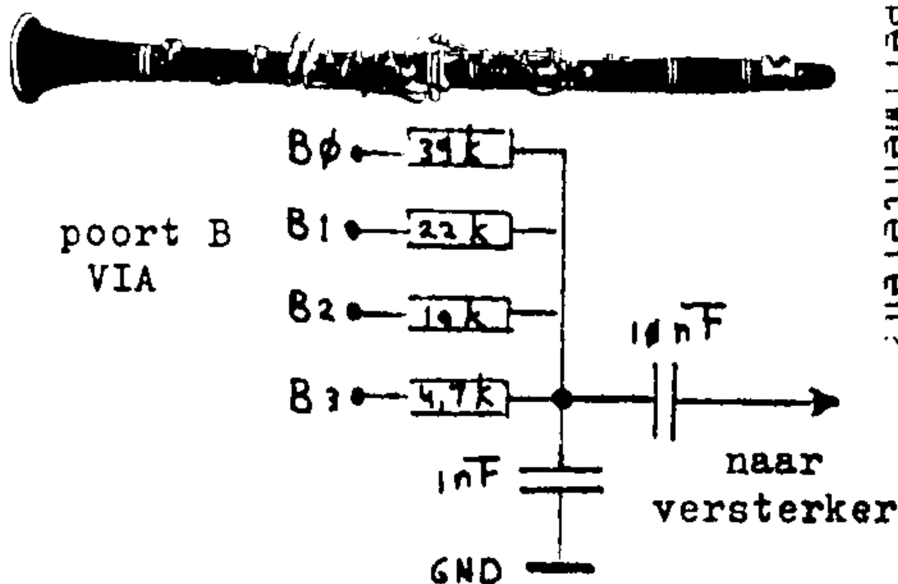
Naar aanleiding van het artikel GOLFVORM uit A.N. 2 83 heb ik geëxperimenteerd met een eenvoudige D/A omzetter (zie schema). Een 4 bits digitaal signaal op poort B van de VIA wordt hiermee omgezet in een analoog signaal met 16 niveaus, niet nauwkeurig, maar voor deze experimenten ruim voldoende.
Het programma genereert eerst 2 zaagtand of driehoeksgolven met verschillende (of gelijke) frequentie. Deze golfvorm wordt vastgelegd in een tabel op #3A00 en #3B00 (r. 1000 tot 1120).

```
   0 REM TOONKUNST
  10 REM GEBRUIKT 4 BITS D/A OMZETTER EN 6522
  20 REM ZIE SCHEMA
 100 DIM LL(5)
 110 FOR I=1 TO 2;DIM P(-1)
 120 P.$21
 130[
 140:LL0 LDY#B1;LDX#B0
 150:LL1 LDA#3B00,Y
 160 CLC;ADC#3A00,X;LSRA
 170 STA#B800;NOP;NOP
 180 DEY;BNE LL2;LDY#B1
 190:LL2 DEX;BNE LL3;LDX#B0
 200:LL3 LDA#B001;AND#80
 210 BNE LL1;RTS
 220]
 230 P.$6;N.
 400 ?#B802=#0F
 410 P.$12
 500aIN."ZAAGTAND (1) OF DRIEHOEK (2) "Q
 510 H=0
 520 IF Q=2;G.b
 530 IF Q=1;IN."ZAAGTAND /(3) OF \(4) "H
 540 IF H=3 OR H=4;G.b
 550 G.a
1000bP."GEEF 2 FREQUENTIES"'
1010 P."TUSSEN 100 EN 6000 HZ"'
1020 FOR K=1 TO 2
1030 E=#3900+256*K
1040 F.K;IN." FREQUENTIE (HZ) "F
1050 B=25600/F;K?#AF=B
1060 C=B/Q;%G=15/C
1070 FOR I=0 TO C
1080 D=%(%G*I)
1090 IF H<>3;E?I=D
1100 IF H=3;E?I=15-D
1110 IF Q=2;E?(B-I)=D
1120 N.;N.
1130 LINK LL0;G.a
```

Tijdens de executie van het machinetaal gedeelte worden de waarden uit de tabel gesommeerd en door 2 gedeeld (r.150,160). Ontsnappen uit het machine programma kan door op SHIFT te drukken (r.200). Experimenteer eens met tabellen van sinusvormige signalen of dit soort golfvormen:



(spraak is zo opgebouwd!). Als aan r.180 wordt toegevoegd: DEY, krijg je bij lage frequenties zwevingen en faser effecten.



Eenvoudige D/A omzetter.

Misschien iets voor synthesizer fanaten, of voor mensen die met computerspraak willen experimenteren?

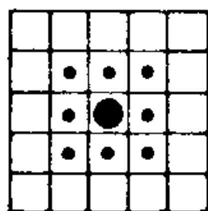
Marien van Westen

LIFE

Het computertijdperk heeft een enorme omwenteling teweeggebracht op het gebied van de behendigheidsspelen. De goede oude flipperkast heeft grotendeels het veld moeten ruimen voor videospelletjes als Space Invaders (zelf alweer een old-timer), Moon Cresta, Scrambler en Pacman. Over het algemeen hebben die spelletjes de naam nogal geestdodend te zijn. Ernstige afstomping zou dreigen voor fanaten die dagen achtereen denkbeeldige ruimtemonsters uiteen doen spatten. Hoe dat ook zij – de computer heeft óók een beeldschermspel mogelijk gemaakt dat veel eenvoudiger is dan het simpelste videospel, dat qua spelregels nauwelijks boven boter kaas en eieren uitstijgt, en waar de speler desgewenst helemáál niets meer bij hoeft te doen, maar waar wiskundigen en computerexperts over de hele wereld nu al meer dan tien jaar gefascineerd onderzoek naar verrichten. In dit spelletje, dat „leven" heet, en dat op een eenvoudige huiscomputer kan worden gespeeld, liggen vragen en misschien zelfs antwoorden besloten over het bestaan van God, de vrije wil van de mens, het groeien en afsterven van organismen en maatschappijen, en het wezen van ruimte en tijd.

„Leven" werd in 1970 bedacht door de Engelse wiskundige Conway. Het spel speelt zich als het ware zélf, onder toezicht van één of meer personen die niets anders mogen doen dan de beginstand bedenken en daarna de regels toepassen. Het „bord" van „leven" is een oneindig raster waarvan de vakjes in twee toestanden kunnen verkeren: leeg of vol. We zullen het uit praktische overwegingen verder hebben over *damstenen* op een oneindig schaak*bord*, maar dat is slechts een voorbeeld dat door een ander vervangen had kunnen worden. Bij het begin („generatie 0") staan op sommige vakjes stenen, in een door de speler te bepalen formatie. Dit is dan ook meteen zijn laatste bemoeienis, want wat er verder generatie na generatie met die stenen gebeurt hangt uitsluitend af van de drie spelregels.

Die regels zijn gebaseerd op het begrip „aangrenzing". De acht vakjes „rond" een steen worden aangrenzend genoemd, de vakjes dus waar hij heen zou kunnen als hij een koning bij schaken was (fig. 1).

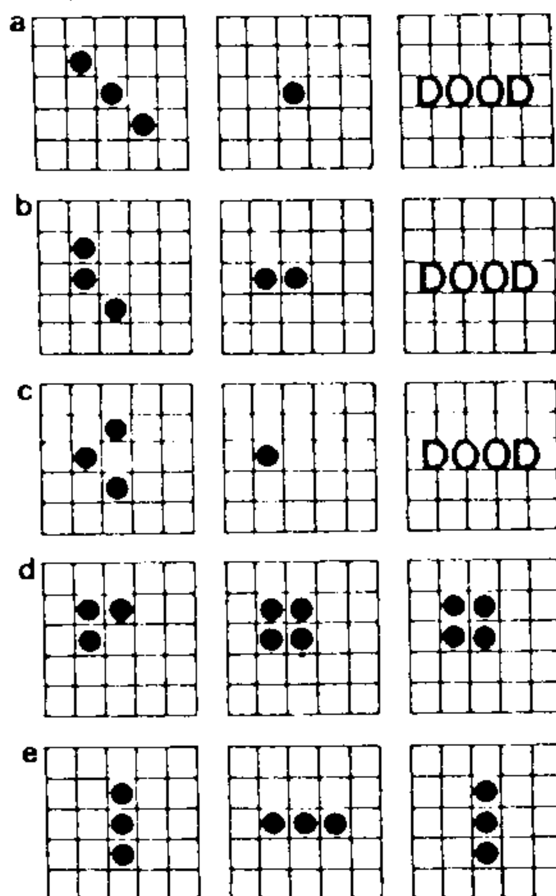


1

Regel 1: Dood. Een steen die aan minder dan twee óf aan meer dan drie stenen grenst „sterft" en wordt verwijderd.
Regel 2: Geboorte. Op een leeg vakje dat aan precies drie vakjes met stenen grenst, verschijnt een nieuwe steen.
Regel 3: Overleven. Een steen die aan twee óf drie vakjes met stenen grenst blijft staan.

NB: Deze drie regels worden steeds *gelijktijdig* toegepast. En zo generatie na generatie, tot in het oneindige.

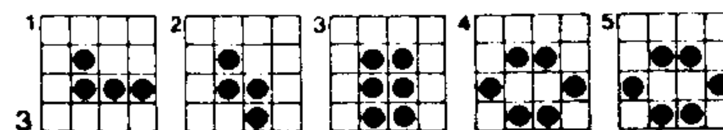
Omdat vooral beginnende „leven"-spelers nogal eens vergissingen maken de volgende tip. Doe het met echte damstenen. Begin met zwarte voor generatie O. Leg tijdens een zet een zwarte steen óp de stenen die zullen sterven, maar laat ze nog even liggen. Leg witte stenen op de „geboorte"-velden. Als alle veranderingen gecontroleerd zijn, verwijder dan de dubbele zwarte stenen, en vervang de witte door zwarte. Op deze manier kunnen een paar van de elementaire „leven"-gebeurtenissen worden bekeken.



2

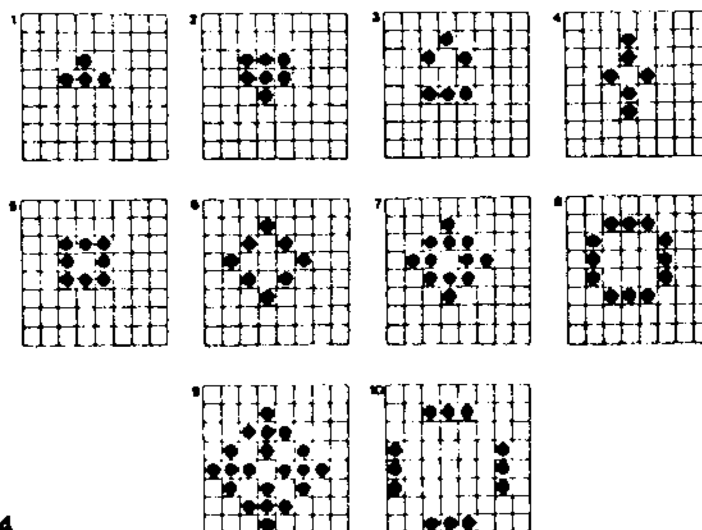
In fig. 2 staat aangegeven wat er met een paar formaties van 3 stenen gebeurt. Het blokje van vier in *d* blijft eeuwig zo staan. *e* is een eeuwige „oscillator" (triller) met fase 2, wat wil zeggen dat hij na twee generaties telkens weer zijn oude gedaante aanneemt.

In fig. 3 ontstaat de „bijenkorf", een verder onbeweeglijk groepje van 6 stenen.



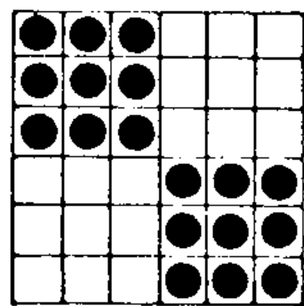
3

In fig. 4 ontstaat het „verkeerslicht".

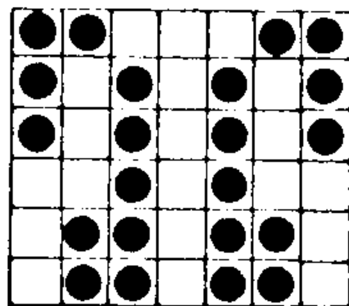


4

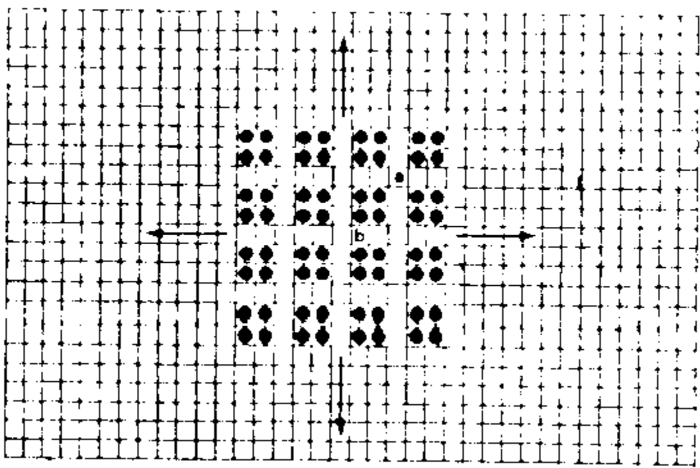
Al spoedig ontdekte men een ware schat aan „stillevens", knipperaars en oscillatoren. Hoe simpel „leven" ook is, daarbij deed zich spoedig het bezwaar voelen dat het construeren van een nieuwe generatie, als er meer dan een paar stenen mee gemoeid waren, ingewikkeld en tijdrovend kon zijn. Beide bezwaren kunnen ondervangen worden met een computerprogramma voor „leven", zoals dat (in BASIC) op pag. 29 staat afgedrukt. Het voordeel van het op een beeldscherm afspelen van een „leven"-configuratie is dat dat met een snelheid kan worden gedaan (bijvoorbeeld 1 seconde per generatie) waarbij de veranderingen als bewegingen worden waargenomen.


5

De „acht" van fig. 5 lijkt op een 8 en is een oscillator met fase 8.

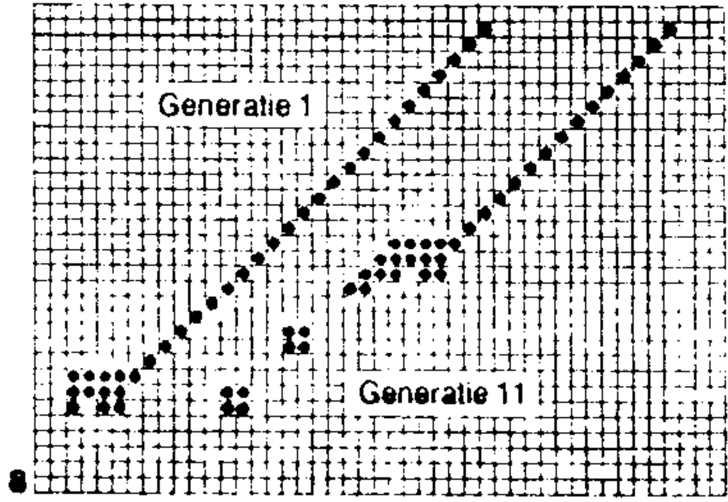

6

De duikelaar van fig. 6 keert zich iedere 7 zetten ondersteboven.


7

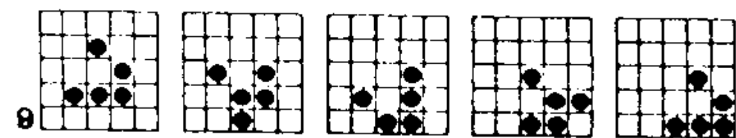
Een vondst was het „virus" (fig. 7). Plaatsen we in een veld van „4-blokken" één nieuwe steen, dan maakt het nogal wat uit waar we dat doen. Op *a* ruimt het veld hem in twee generaties spoorloos op, en het blijft daarna eeuwig zo bestaan. Maar op *b* (en op alle symmetrisch overeenkomstige velden) is het verhaal omgekeerd. Op gruwelijke wijze wordt dan het hele veld vanuit het midden weggevreten; na 5 generaties * is er niets meer van over. (Zie beeldschermfoto's.)

*: herstel: 32 generaties

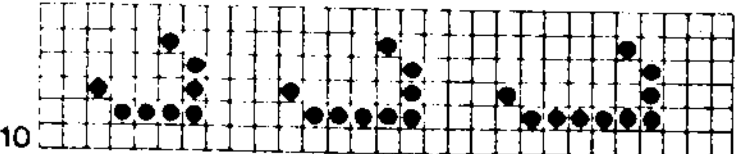


8

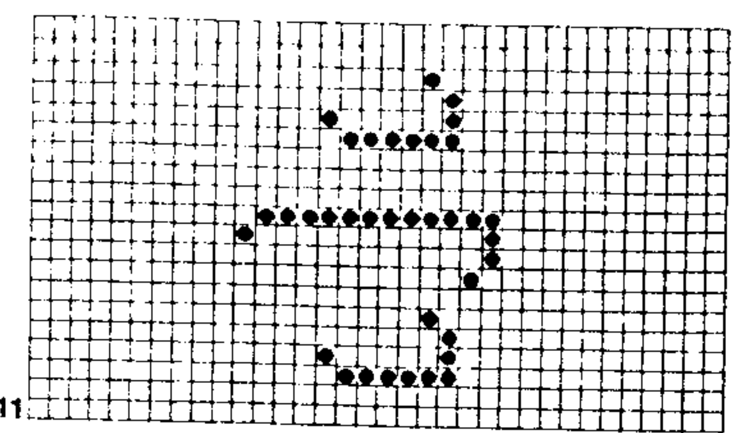
In fig. 8 kruipt een „hooimachine" langs een oneindige diagonaal van stenen omhoog, en hij laat ze netjes in „schoven" van 4 achter. Duidelijker en eenvoudiger nog is de beweging in fig. 9.


9

De groep van vijf, „zwever" genaamd, verplaatst zich op eigen houtje iedere 4 generaties één vakje naar rechtsonder. Eeuwig zal hij doorvliegen. Dat doet aan een ruimteschip denken, maar hij is lang niet het enige. De formaties in fig. 10 reizen per vier zetten steeds twee velden naar rechts, en laten daarbij steeds een „vonk" in de lucht achter die na één zet verdwijnt.


10

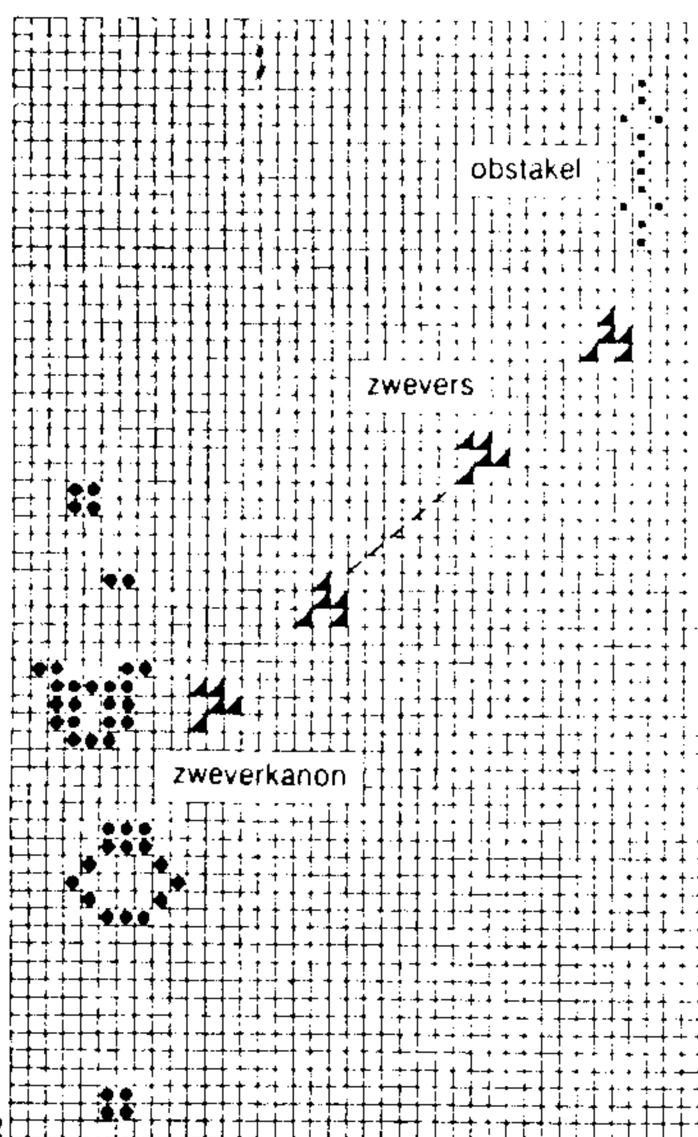
Grotere ruimteschepen zijn door de grotere rommel die ze achterlaten niet mogelijk (die rommel gaat het schip aantasten – als dat eens met uitlaatgassen kon!), maar topspelers ontdekten dat het wél gaat als we ze laten escorteren door kleinere ruimteschepen. In fig. 11 staat zo'n konvooi.


11

Iemand construeerde zelfs een ruimteschip van 100 stenen, dat geëscorteerd door een vloot van 33 kleinere ruimteschepen, eeuwig over het oneindige schaakbord verder vloog.

inderdaad fascinerende mogelijkheden en het bewijs dat Conway zijn drie regels goed bedacht had. Want „leven" is geen toevalstreffer geweest al zou men dat door de eenvoud van die regels denken – pas na lang proberen vond Conway de goede regels die de configuraties niet te snel lieten uitsterven of aangroeien. Dat zou het spel oninteressant of onoverzichtelijk hebben gemaakt. De gelijkenis die met het leven zélf ontstaat is evenmin toevallig, maar juist Conway's bedoeling. Sommige groepen zijn levensvatbaar, andere stabiel, weer andere sterven uit – net als menselijke culturen.

Werkelijk fantastisch werden de ontdekkingen naar aanleiding van een weddenschap van Conway, overigens slechts om 50 dollar, maar het ging natuurlijk om de eer. Conway veronderstelde aanvankelijk dat geen enkel patroon eeuwig kon groeien. Een groep wiskundigen van het beroemde MIT, het Massachusetts Institute of Technology in Boston in Amerika, liet zien dat dit wel degelijk mogelijk is. Ze construeerden het „zwever-kanon" van fig. 12.

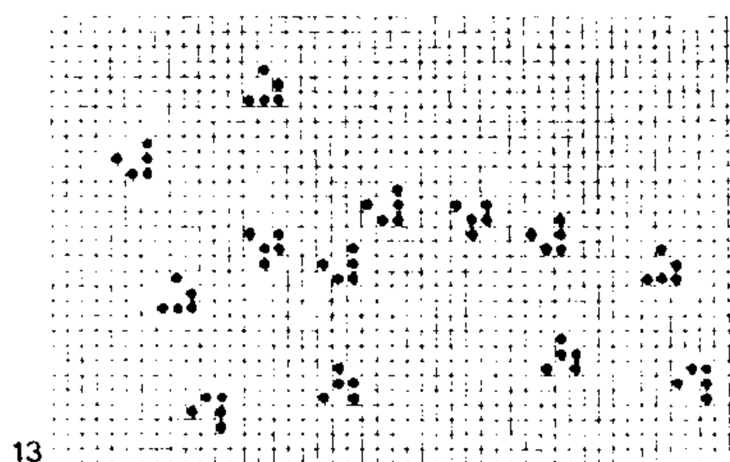


12

Dit is een oscillator met fase 30, die tijdens iedere cyclus één zwever afvuurt. Er komen dus eeuwig groepjes van 5 bij. Conway was zijn 50 dollar kwijt en een enorme aanwinst voor zijn „leven" rijker.

Dat eeuwige aangroeien gaat overigens niet door als op de juiste plek een „eter" wordt opgesteld. Dat is een oscillator met fase 15 die ondertussen zwevers verzwelgt zonder dat er iets van te merken is. (Zie de beeldschermfoto's.)

Er zijn ook „pingpong-batjes" die zwevers eeuwig naar elkaar toe kunnen kaatsen en prachtige effecten kunnen ontstaan wanneer zwevers en/of ruimteschepen met elkaar botsen. Soms blijft er dan in het geheel niets over. Maar je kan een paar zwever-kanonnen ook zo opstellen dat hun botsende zwevers samen een fabriek bouwen waaruit om de 300 generaties een ruimteschip wordt gelanceerd. Soms vormen die zwevers samen een kanon dat op het oorspronkelijke kanon begint te schieten. Fig. 13 is van iets dergelijks een betrekkelijk eenvoudig voorbeeld.


13

Er vliegen hier 13 zwevers op elkaar af die bij generatie 75 een zwever-kanon hebben gevormd dat vervolgens om de 30 generaties een zwever afvuurt.

Hoe simpel de regels van „leven" ook zijn – er zit zoveel in dat wiskundigen nu, meer dan tien jaar na het bedenken van het spel, nog bezig zijn de raadselen en mogelijkheden ervan op te lossen. Er heeft zelfs enige tijd een internationaal tijdschrift voor „leven" bestaan.

Het spel appelleert niet alleen aan het gevoel voor wiskundige schoonheid, maar ook aan een paar diep-menselijke gevoelens. Zo heeft het feit dat iedere generatie geheel bepaald wordt door de vorige en door niets anders een sterke overeenkomst met de natuurwetten – ook daar kan iedere nieuwe situatie in principe geheel berekend worden uit de vorige. Kan de vrije wil van de mens dan bestaan? Hij wordt immers geregeerd door natuurwetten? Het antwoord is dat „vrije wil" uitsluitend bestaat in de ogen van de „willer". Zolang je voelt dat je een vrije wil hebt, heb je die. Volgende vraag: kan dan ook een „zwever" van Conway uit vrije wil zweven of zo'n hooimachine hooien?

Wie over die vraag doordenkt zal een verrassende ontdekking doen. Conway's „leven" is een uitermate simpel spel. Zonder aan het wezen van dat spel iets te veranderen kunnen we het ingewikkelder maken. Met de computer zijn nooit configuraties van meer dan hooguit een paar honderd stenen onderzocht. Maar zet er eens tien miljard (het aantal zenuwcellen in het menselijk brein) neer, of nog veel meer. Neem in plaats van een schaakbord een *drie*-dimensionaal raster, en definiëer in plaats van 8 „aangrenzende" vakjes honderd of een paar duizend als zodanig (dat laatste is het aantal verbindingen van de gemiddelde menselijke zenuwcel). Neem tenslotte in plaats van twee, *honderd* toestanden waarin die vakjes kunnen verkeren.

Dat het spel zich dan oneindig veel ingewikkelder presenteert zal duidelijk zijn. Het is onmogelijk te voorspellen wat er dan gebeurt. Zouden we op die manier niet een soort brein hebben geschapen dat alleen maar hoeft te worden aangezet om eeuwig door te blijven krioelen? In wat zou dat gekrioel dan eigenlijk nog te onderscheiden zijn van het gekrioel van elektronen in onze hersenpan? Zou er dan niet sprake zijn van een wezen dat echt leeft, en zou de bedenker van de regels waar het aan gehoorzaamt dan niet God moeten worden genoemd? Zou zo'n wezen niet wanhopig bezig kunnen zijn met een poging te ontdekken *hoe* die regels eruit zien waarnaar hij luistert, én naar degeen die ze bedacht heeft?

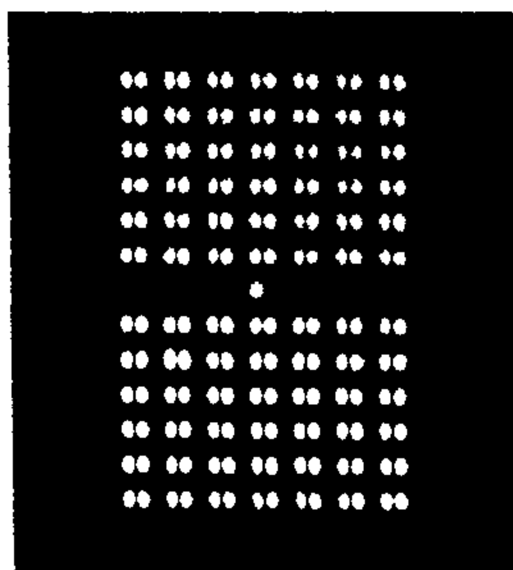
Met andere woorden: wie zegt dat wij mensen niet een rommeltje van „damstenen" zijn die op het raster van een of ander „leven"-spel aan onduidelijke regels gehoorzamen?

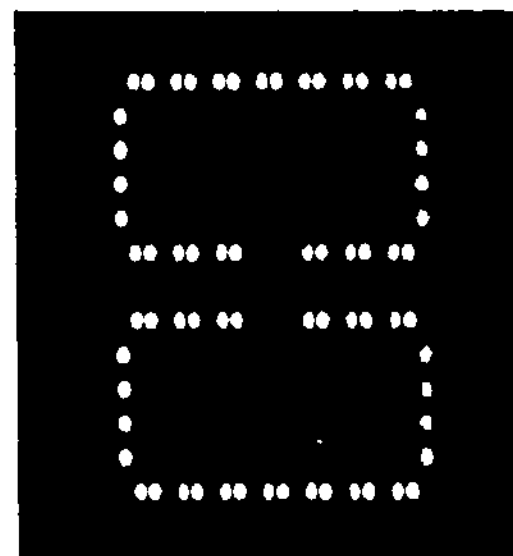
Een van de belangrijkste ontdekkingen van Conway's „leven" is misschien dat tegen deze redenering niets in te brengen is. Een tegenwerping zou kunnen zijn dat volgens ons gevoel tijd en ruimte vloeiende zaken zijn tussen de delen waarvan geen grenzen bestaan. Maar net zoals men heeft ontdekt dat de energie op basisniveau uit quantums bestaat, zo is het heel goed denkbaar dat dit met tijd en ruimte ook het geval is. De mens is heel goed voorstelbaar als iets dat een (groot) aantal hokjes vult – een discontinuë ruimte inneemt. Beweegt hij, betekent dat niets anders dan dat er andere vakjes gevuld worden. Zo'n beweging gaat dan met een sprong – zoals de wijzers van sommige klokken, of de beeldjes van een film dat doen. Als die sprongen klein zijn, zou aan niets te merken zijn dat de tijd geen vloeiende beweging is, maar een discontinuë.

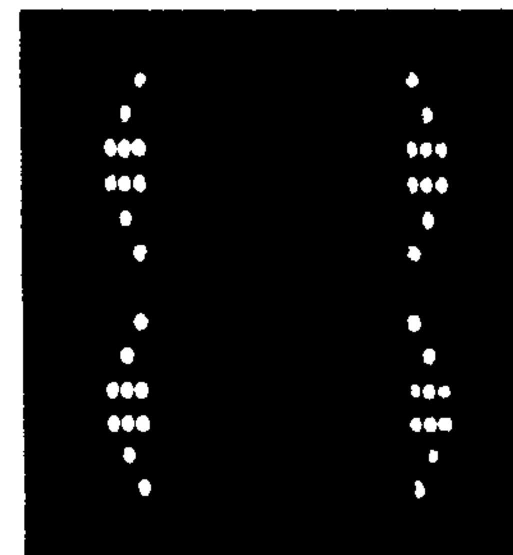
Ook voor tijd en ruimte dus een quantumtheorie? Het zou schitterend zijn als een veredeld boter, kaas en eieren zo'n revolutionair nieuw inzicht zou hebben veroorzaakt.

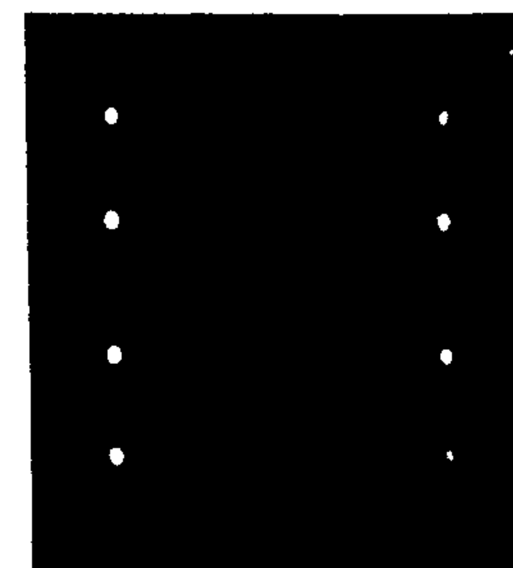
Tekst: Robert Rijs

LITERATUUR
Manfred Eugen / Ruthild Winkler. Laws of the game, Alfred A. Knopf, New York 1981









Foto's: Dominique van Rossum

Dit artikel werd, met welwillende toestemming van de uitgever, overgenomen uit het maandblad "KIJK", van November 1983.
Onze hartelijke dank hiervoor.

Het oorspronkelijk bij dit artikel behorende computer-programma is weggelaten en vervangen door het betere en snellere programma van Peter Ehrlich, op de volgende blz.

```
  1REMconway transformatie
  2REM PROGRAMMA V. P.EHRLICH
 10 DIM A1190,B1190,C176
 11 DIM LL26,P(-1)
 20 F.N=0TO26;LL(N)=P;N.;P.$21
 30 Q=P;GOS.b;P=Q;GOS.b;P.$6
 40 F.N=A TO A+1190 S.4;!N=0;N.
 45 F.N=B TO B+1190 S.4;!N=0;N.
 50 P.$30;CLEAR0
 60 P."GEBRUIK TOETSEN A Z < >"
 70 P.'"AAN/UIT MET SPATIE; BEGIN=B"
100 X=32;Y=25;PLOT14,X,Y
110 DO LINK LL26
120 IF ?#8D<>#20;G.a
130 PLOT14,X,Y;Z=X-15+(42-Y)*34
140 A?Z=A?Z:1
150aIF?#8D=#41;IF Y+1<42;PLOT14,X,Y;Y=
Y+1;PLOT14,X,Y
160 IF?#8D=#5A IF Y-1>8;PLOT14,X,Y;Y=Y
-1;PLOT14,X,Y
170 IF?#8D=#2E IF X+1<48;PLOT14,X,Y;X=
X+1;PLOT14,X,Y
180 IF?#8D=#2C IF X-1>15;PLOT14,X,Y;X=
X-1;PLOT14,X,Y
190 UNTIL ?#8D=#42
200 !#80=A;!#82=B;CLEAR0
210 DO !#8A=#8048;LINK LL9;U.0 .
300b[
310:LL0 LDA#80;STA#84;LDA#81;STA#85
320 LDA#82;CLC;ADC@35;STA#86
330 LDA#83;ADC@0;STA#87;RTS
340:LL5 LDX@0;LDY@0;LDA@3;STA#8C
350:LL1 LDA(#84),Y;BEQ LL2;INX
360:LL2 INY;LDA(#84),Y;BEQ LL3;INX
370:LL3 INY;LDA(#84),Y;BEQ LL4;INX
380:LL4 CLC;TYA;ADC@32;TAY
390 DEC#8C;BNE LL1;SEC;TYA
400 SBC@67;TAY;LDA(#84),Y;BEQ LL6
410 CPX@5;BPL LL7;CPX@3;BMI LL7;BPL LL
8
420:LL6 CPX@3;BEQ LL8
430:LL7 LDA@0;LDY@0;STA(#86),Y;RTS
440:LL8 LDA@1;LDY@0;STA(#86),Y;RTS
500:LL9 JSR LL0;LDA@33;STA#89
```

```
510:LL10 LDA@32;STA#88
520:LL11 JSR LL5;LDA#84;CLC;ADC@1
530 STA#84;LDA#85;ADC@0;STA#85
540 LDA#86;CLC;ADC@1;STA#86;LDA#87
550 ADC@0;STA#87;DEC#88;BNE LL11
560 LDA#84;CLC;ADC@2;STA#84;LDA#85
570 ADC@0;STA#85;LDA#86;CLC;ADC@2
580 STA#86;LDA#87;ADC@0;STA#87
590 DEC#89;BNE LL10;BEQ LL19
600:LL12 LDY@0;LDX@#40;LDA(#86),Y
610 BEQ LL13;TXA;ORA@#20;TAX
620:LL13 INY;LDA(#86),Y;BEQ LL14
630 TXA;ORA@#10;TAX
640:LL14 TYA;CLC;ADC@33;TAY;LDA(#86),Y
650 BEQ LL15;TXA;ORA@8;TAX
660:LL15 INY;LDA(#86),Y;BEQ LL16
670 TXA;ORA@4;TAX
680:LL16 TYA;ADC@33;TAY;LDA(#86),Y
690 BEQ LL17;TXA;ORA@2;TAX
700:LL17 INY;LDA(#86),Y;BEQ LL18
710 TXA;ORA@1;TAX
720:LL18 TXA;RTS
730:LL19 JSR LL0;LDA@11;STA#89
735 LDX@0;STX#8C
740:LL20 LDA@16;STA#88
750:LL21 JSR LL12;LDX#8C;STA C,X
755 INX;STX#8C
760 LDA#86;CLC;ADC@2;STA#86
770 LDA#87;ADC@0;STA#87;DEC#88
780 BNE LL21;LDA#86;CLC;ADC@70
790 STA#86;LDA#87;ADC@0;STA#87
810 DEC#89;BNE LL20
820 LDA#80;LDX#82;STX#80
830 STA#82;LDA#81;LDX#83;STA#83
840 STX#81;LDX@0;LDA@11;STA#89
850:LL23 LDA#B002;BMI LL23
860:LL24 LDY@0
870:LL25 LDA C,X;STA(#8A),Y
880 INX;INY;CPY@16;BNE LL25
890 LDA#8A;CLC;ADC@32;STA#8A
891 LDA#8B;ADC@0;STA#8B
895 DEC#89;BNE LL24;RTS
900:LL26 JSR#FE94;STA#8D;RTS
910];R.
>
```

** SOURCEMAKER **

HET GEBRUIK VAN DE SOURCEMAKER

INLEIDING

In het algemeen wordt een programma in machine-code via een hulpprogramma (in Basic) aangemaakt. Dit hulpprogramma wordt ook wel Source-programma genoemd. De assembler zorgt er dan voor, dat de mnemonics (zoals b.v. STA of LDA) in machine-code wordt omgezet. Deze code kan met behulp van een disassembler-programma op eenvoudige manier worden teruggelezen. Willen we de code veranderen, dan zullen de veranderingen via de Source moeten worden aangebracht. Wanneer we echter niet de beschikking hebben over het Source-programma, dan wordt het problematisch.
We kunnen dan het Source-programma maken aan de hand van een disassembler listing. Er is echter een veel eenvoudiger manier:

Het programma - - - - SOURCEMAKER

WERKING

Laden programma

Het prgramma dient te worden geladen via *LOAD"SOURCEMAKER"
Het programma is zodanig ontworpen dat het zowel in het hoge geheugengebied als ook in het lage geheugengebied kan worden gerund. Het mag dus in b.v. #8200, maar ook in #2900 worden geladen.

Starten programma

Nadat het programma is geladen, moet de textpointer worden verplaatst naar een waarde 2 hoger dan waar het programma werd ingeladen. Dus is het programma geladen in #8200, dan moet de textpointer naar #84.
Het programma werkt conversationeel. Dat wil zeggen dat aan de hand van een aantal vragen het programma bepaald welke uitvoering gewenst is. Na het RUN-commando is de eerste vraag die wordt gesteld:

STARTADRES ?

Hier dient het beginadres van de machine-code te worden ingevuld. Dit adres mag met of zonder #-teken worden ingevoerd. Bij ongeoorloofde karakters volgt de melding "INPUT FOUT". Het startadres kan dan opnieuw worden ingetikt. Bijvoorbeeld:

STARTADRES ? A002

Dan volgt de vraag:

EINDADRES ?

Het adres moet op dezelfde wijze worden ingevoerd als bij de eerste vraag, alleen mag hierbij direct een return worden ingetikt. Wordt op deze vraag een return gegeven, dan zal het programma verder alleen nog als disassembler funktioneren.
Is het eindadres opgegeven dan is de volgende vraag:

ADRES VOOR OPSLAG SOURCE ?

Hier voeren we het adres in waar we de source willen hebben. Ook hier kunnen we volstaan met een return, waarna het programma alleen als disassembler werkt.

Opmerking: Wanneer het opgegeven adres binnen het programma valt, dan volgt de melding: GEEN RUIMTE VOOR OPSLAG SOURCE !
Het programma stopt hierna.

Hebben we bij bovenstaande vraag wel een adres opgegeven, dan volgt:

GEEF SOURCENAAM ?

Deze vraag spreekt voor zich.
Dan krijgen we nog 2 vragen, die ook worden gesteld wanneer we het programma alleen als disassembler gebruiken.

PAGE MODE J/N ?

PRINTER J/N ?

Ook deze vragen behoeven geen verdere toelichting.

Gebruiken we het programma als disassembler dan volgt de melding:

pass 1

Hierna volgt dan de gedisassembleerde code over het scherm. De opties (PAGE MODE en PRINTER) zijn nu actief.
Wanneer er geen eindadres was opgegeven, dan loopt het progremma door tot ca. 31 februari. In het andere geval stopt het programma bij het bereiken van het eindadres.

Opmerking: Het disassembleren kan d.m.v. de shift-toets worden afgebroken. Het programma komt dan weer met de vraag "STARTADRES".

Zijn we daarintegen bezig met de aanmaak van een source-programma dan vervolgt het programma na "PRINTER J/N" met:

EIGEN LABELS INVULLEN:
LABEL 1 ADRES ?

We kunnen nu adressen invullen, die later in de source van een label moeten worden voorzien. Deze loop kan worden gestopt d.m.v. een return. Hierna vervolgt het programma met "pass 1" en is de gedisassembleerde code op het scherm te zien.

Opmerking: Wanneer het aantal labels groter is dan 250, volgt de melding: TEVEEL LABELS !
Het programma stopt dan.

Bij het bereiken van het eindadres volgt de melding:

pass 2
WACHT....

Het programma is nu bezig met de aanmaak van de source. Dit kan, afhankelijk van de grootte van de code, enkele seconden tot tientallen minuten duren.

Opmerking: Tijdens de aanmaak van de source kan het programma worden afgebroken met de melding: GEEN RUIMTE VOOR OPSLAG SOURCE !
Het eventueel aangemaakte stuk programma is NIET te gebruiken.

Is het aanmaken van de source zonder problemen verlopen, dan stelt het programma tot slot de vraag:

TABEL MAKEN J/N ?

Indien we bij deze vraag een return of N intikken dan stopt de SOURCEMAKER.
In het andere geval kunnen we nu in het Source-programma een tabel opnemen. Dit kan bijvoorbeeld Data zijn die voor of achter de machinecode ligt. Het begin- en eindadres van dit stuk Data geven we dan op bij de vragen "STARTADRES" en "EINDADRES". De Sourcemaker stopt dan met de melding:

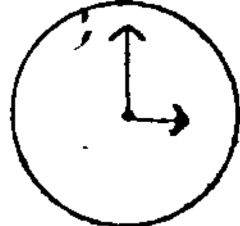
klaar

We kunnen nu de textpointer verplaatsen naar het geheugengebied waar de Source is aangemaakt.

VEEL SUCCES . . . .

m.v.g.
A.Marchal E.Konda

W E R E L D T I J D



```
10 REM WERELDTIJD
20 REM AUTEUR: CHRISTIAN BOHN
30 REM UIT CHIP JUNI 1981
40 REM BEWERKT VOOR ATOM DOOR W ERNST, ECHT
50 PRINT $12;Z=0;e=3;F=10000;PRINT'''''
60 PRINT"          ++++++++++++++++++"'
70 PRINT"          +++ WERELDTIJD +++"'
80 PRINT"          ++++++++++++++++++"''
85 FOR N=1 TO 30; WAIT; NEXT N
90 IF Z=1 THEN GOTO 240
100 DIM P(20),Q(1);Z=1
110 INPUT'''"PLAATS (IN NEDERLAND)"$P
115 INPUT"ZOMER- OF WINTERTIJD Z/W)"$Q
120 PRINT"GEEF DE JUISTE TIJD AAN   ";INPUT"H,M,S"T
125 IF $Q="Z"; T=T-10000
130 REM SAN FRANSISCO
140 A=T-90000;IF A<O THEN A=A+240000
150 REM NEW YORK
160 B=T-60000; IF B<O THEN B=B+240000
170 REM MOSKOU
180 C=T+20000;IF C>240000 THEN C=C-240000
190 REM BOMBAY
200 D=T+20000; IF D>240000 THEN D=D-240000
210 REM MELBOURNE
220 E=T+90000;IF E>240000 THEN E=E-240000
230 IF $Q="Z"; T=T+10000
235 PRINT $12;GOTO 60 30
240aPRINT'''$P;GOSUB 370  B
250 PRINT T/F,(T%F)/100,T%100'
260 PRINT"SAN FRANSISCO";GOSUB 370
270 PRINT A/F,(A%F)/100,A%100'
280 PRINT"NEW YORK";GOSUB 370
290 PRINT B/F,(B%F)/100,B%100'
300 PRINT"MOSKOU";GOSUB 370
310 PRINT C/F,(C%F)/100,C%100'
320 PRINT"BOMBAY";GOSUB 370
330 PRINT D/F,(D%F)/100,D%100'
340 PRINT"MELBOURNE";GOSUB 370
350 PRINT E/F,(E%F)/100,E%100''
360 PRINT"DRUK TOETS";LINK#FFE3;RUN
370bDO PRINT" ";UNTIL COUNT=20;RETURN
```

Mensen die in het bezit zijn van een MONITOR of een tot monitor omgebouwde T.V. en vinden dat de LICHT OPBRENGST onvoldoende is, kunnen het volgende doen.

Op de hoofdprint (computer) LINK5 doorkrassen. Pen 39 van de 6847 via een weerstand van ± 4700 Ohm met de + 5 Volt verbinden.

Degene die een printer hebben en deze LINK reeds doorgekrast, hebben pen 39 reeds aan massa gelegd. Door deze verbinding te vervangen door een weerstand van 4700 Ohm naar de + 5 Volt wordt de lichtopbrengst met ca. 50 % verhoogd.

C O M P L E X - R E K E N E N

'n wiskunde hulpje

```
4 REM COMTLEX REKENEN
5 REM VOOR ACORN ATOM BEWERKT DOOR C EN W CAMPERS ROERMOND
6 REM DATUM 1-8-1982
10 P.$12"COMPLEX REKENEN"'"================"''$7
20 FIN."EERSTE COMPLEXE GETAL - REEEL GEDEELTE "%A;P.'$7
25 FIN."IMAGINAIR GEDEELTE "%B;P.'$7
30 FIN."TWEEDE COMPLEXE GETAL - REEEL GEDEELTE "%C;P.'$7
40 FIN."IMAGINAIR GEDEELTE "%D
45 %Z=((%A*%C)-(%B*%D)); %Y=((%B*%C)+(%A*%D))
50 %W=((%A*%C)+(%B*%D))/((%C*%C)+(%D*%D))
51 %V=((%B*%C)-(%A*%D))/((%C*%C)+(%D*%D))
52 P.$12$7$7$7"UITKOMSTEN"'"============"'"Z1      ="'
71 FP."  "%A;FIF %B>=0 THEN P."+"
75 FP.%B"J"'"Z2      ="'"  "%C;FIF %D>=0 THEN P."+"
85 FP.%D"J"'"Z1+Z2 ="'"  "%A+%C;FIF %B+%D>=0 THEN P."+"
122 FP.%B+%D"J"'"Z1*Z2 ="'"  "%Z;FIF %Y>=0 THEN P."+"
132 FP.%Y"J"'"Z1/Z2 ="'"  "%W;FIF %V>=0 THEN P."+"
142 FP.%V"J"'"================================"''
155 IN."NOG EEN BEREKENING J/N "R; IF R=N G.190
170 IF R=J P.$12;%A=0;%B=0;%C=0;%D=0;G.10
175 G.155
190 END
```

T.E A - C U P S

'n spelletje

```
0P.$12"shift=LEFT  rept=RIGHT"''"WAIT";F.N=0TO200;WAIT;N.
1CLEAR0;S=0;C=0;?#E1=0
2P.$30"     CUPS        SCORE"'
4K=33260;?K=255
8F.C=0TO50
10P=A.R.%32+32832
20F.T=0TO13;L=T*32+P
21IF?#B002&64=0;K=K+1;?K=255;?(K-1)=192;IF K=33279;K=K-1
22IF?#B001&128=0;K=K-1;?K=255;K?1=192;IF K=33248;K=K+1
23?33248=192;?33279=192
25WAIT;WAIT;WAIT
30?L=123;?(L-32)=192
35IF L>33247;?L=192
40N.
45IF K=L;P.$7;S=S+1;?32790=S/10+48;?32791=S%10+48
46?32778=C/10+48;?32779=C%10+48
48N.;E.
```



Mensen die het gemis van een AUTO-REPEAT functie van de toetsen als een lacune ervaren, kunnen dit verhelpen door een draadbrug over de REPT-toets te solderen. Het is echter wel even wennen!!!

TIPS van John van Schaijik.

----Op een van de Club-avonden hoorde ik dat er geen foutloos werkende afrondings-routine was; sommige getallen o.a. 2,345 zouden problemen geven. Hierbij een listing van een programma'tje waar van ik nog geen fout gevonden heb. Listing 1.

----Voor degene die met een Disk-drive werken is misschien het volgende van belang. Het commando *VDU maakt het scherm schoon (misschien nog meer?).

Met het commando *SPOOL is het mogelijk een FILE die ge-DELETED is weer terug te krijgen. U typt na abusievelijk wissen *SPOOL"FILENAME" (quotes mogen weg) en hup een ramp is weer voorkomen.

Het aandruk-viltje in de Drive blijkt nog al eens problemen te geven (Dank aan J.Vogten voor de tip). Het laat gemakkelijk los. U kunt met een zaklamp van buitenaf zien of het nog aanwezig is.

Het programma ATOM-STORE geeft bij gebruik van een D-Drive problemen om de DATA weer terug van schijf te krijgen. Hierbij een laad-routine die alle problemen oplost. Listing 2.

Roermond 13.08.1983 Met vriendelijke groet,
John van Schaijik.

LISTING 1

```
    1 REM SUBROUTINE "SCHRIJVEN
      ACHTER DE KOMMA"
    2 FIN.%A;GOSUB a
   20 END
10000a @ =0
10005 A=0;B=0;C=0;D=0
10010 %B=%A*1000;A=%B
10020 B=A/1000
10030 C=1000*B
10040 D=A-C;D=D+5;D=D/10
10050 P.B,",",D;R.
```

LISTING 2

```
 20 P.$12
 25 P."adressen-bestand"''
 30 P."INSERT FLOPPY NO.1"'
 32 P."ALS KLAAR DRUK OP SPATIE-
    TOETS"
 35 LINK#FFE3
 50 P.$21
 60[;JSR #E000;]
 65 P.$6
 66 *LO.STDAT 2800
 70 *LO."STORE" 8200.
 72 P.'"EVEN GEDULD"
 75 F.Q=1TO100;WAIT;N.
 80 ?18=#82
100 RUN
```

DE GEHEUGENKAART en zijn WEERSTANDEN

R1, R2	2K2	R29	220 Ohm
R3 t/m R21	27K	R30	470 Ohm
R22 t/m R27	33K	R31	56 Ohm !!! persé geen 56K
R28	27K kan ook weg	R32	33K

TABULATOR

Deze routines verzorgen een horizontale en vertikale tabulator. Zowel op het beeldscherm, als op de printer!! Ze MOETEN opgenomen worden in een 'toolkit' of 'schakelsoft'. Formaat: HTAB x of VTAB x, waarbij x een getal is, dat aangeeft waar het printen begint. Als argument (x) is ook zondermeer een variabele of zelfs een wiskundige uitdrukking mogelijk.

```
  0 REM  HTAB; VTAB
 10 DIM LL(2); INPUT"ASSEMBLE TO"A
 20 FOR N=0 TO 2; LL(N)=A; NEXT N
 30 FOR N=1 TO 2; P=A;[
 40 \ HTAB
 50 JSR≠C4E1;DEX;STX 4;LDY ≠16,X;BEQ LL1;DEY
 60:LLO JSR≠F379;DEY;BNE LLO
 70:LL1 JMP≠C55B
 80 \ VTAB
 90 JSR≠C4E1;DEX;STX 4;LDY ≠16,X;BEQ LL1;DEY
100:LL2 JSR≠FFED;DEY;BNE LL2;BEQ LL1;JMP≠C55C;]
110 NEXT N;@=0;PRINT"CODE VAN ≠"&A" TOT ≠"&P-1';@=8;END
```

PAUZE

Deze routine maakt een wachttijd mogelijk van max. 4,25 sec. Formaat: PAUZE x; waarbij x niet groter mag zijn dan 255 en niet kleiner dan 0. Ze MOET ook weer gebruikt worden in een 'toolkit' of 'schakelsoft'.

```
  0 REM PAUZE
 10 DIM LL(2);INPUT"ASSEMBLE TO"A
 20 FOR N=0 TO 2; LL(N)=A; NEXT N
 30 FOR N=1 TO 2; P=A;[
 40 JSR≠C4E1; DEC4; LDX4; LDA≠25,Y; TAY; INY; LDA≠16,X; TAX; BEQLL1
 50:LLO JSR≠FB83
 60:LL1 DECO; BEQLLO; JMP≠C55B;]
 70 NEXT N;@=0;PRINT"CODE VAN ≠"&A" TOT ≠"&P-1';@=8;END
```

ROUTINE 1

Deze routine maakt het scherm schoon vanaf de cursor. De regel waar de cursor staat wordt ook schoongemaakt.

```
 10 DIMLL(2);F.N=0 TO 2;LL(N)=≠FFFF;N.
 20 IN."OBJ.-CODE"A;F.N=1TO2;P=A;[
 30 LDA@11;JSR≠FFF4;LDA≠DE;STA≠91;LDA≠DF;STA≠92;JMP LL1
 40:LLO LDY@0;LDA ≠20;STA(≠91),Y
 50:LL1 CLC;LDA≠91;ADC@1;STA≠91;LDA≠92;ADC@0;STA≠92;CMP ≠82
 60 BEQ LL2;JMP LLO
 70:LL2 RTS
 80];N.
 90 REM DEMONSTRATIE
100 P.$30;F.N=1TO5;P.$10;N.;LINK A;END
```

```
 1 REM MEMORY  VOOR 2 PERSONEN
 2 REM DOOR: J.VERSTELLE
 3 REM         SLEUTELBLOEMZOOM 1
 4 REM         2353 RA LEIDERDORP
 10 DIM AA(48),M(10),N(10);@=Ø;I=32;K=Ø;R=Ø
 20 P.$12,"                    memory";?≠E1=Ø;GOS.x
 40 FOR B=1 TO 48;AAB=Ø;NEXT B
 70 FOR C=1 TO 2
 80 FOR D=129 TO 152
 90 E=(ABSRND%48+200);GOTO E
200bIF AA1=Ø;AA1=D;GOTOa
201 IF AA2=Ø;AA2=D;GOTOa
202 IF AA3=Ø;AA3=D;GOTOa
203 IF AA4=Ø;AA4=D;GOTOa
204 IF AA5=Ø;AA5=D;GOTOa
205 IF AA6=Ø;AA6=D;GOTOa
206 IF AA7=Ø;AA7=D;GOTOa
207 IF AA8=Ø;AA8=D;GOTOa
208 IF AA9=Ø;AA9=D;GOTOa
209 IF AA10=Ø;AA10=D;GOTOa
210 IF AA11=Ø;AA11=D;GOTOa
211 IF AA12=Ø;AA12=D;GOTOa
212 IF AA13=Ø;AA13=D;GOTOa
213 IF AA14=Ø;AA14=D;GOTOa
214 IF AA15=Ø;AA15=D;GOTOa
215 IF AA16=Ø;AA16=D;GOTOa
216 IF AA17=Ø;AA17=D;GOTOa
217 IF AA18=Ø;AA18=D;GOTOa
218 IF AA19=Ø;AA19=D;GOTOa
219 IF AA20=Ø;AA20=D;GOTOa
220 IF AA21=Ø;AA21=D;GOTOa
221 IF AA22=Ø;AA22=D;GOTOa
222 IF AA23=Ø;AA23=D;GOTOa
223 IF AA24=Ø;AA24=D;GOTOa
224 IF AA25=Ø;AA25=D;GOTOa
225 IF AA26=Ø;AA26=D;GOTOa
226 IF AA27=Ø;AA27=D;GOTOa
227 IF AA28=Ø;AA28=D;GOTOa
228 IF AA29=Ø;AA29=D;GOTOa
229 IF AA30=Ø;AA30=D;GOTOa
230 IF AA31=Ø;AA31=D;GOTOa
231 IF AA32=Ø;AA32=D;GOTOa
232 IF AA33=Ø;AA33=D;GOTOa
233 IF AA34=Ø;AA34=D;GOTOa
234 IF AA35=Ø;AA35=D;GOTOa
235 IF AA36=Ø;AA36=D;GOTOa
236 IF AA37=Ø;AA37=D;GOTOa
237 IF AA38=Ø;AA38=D;GOTOa
238 IF AA39=Ø;AA39=D;GOTOa
239 IF AA40=Ø;AA40=D;GOTOa
240 IF AA41=Ø;AA41=D;GOTOa
241 IF AA42=Ø;AA42=D;GOTOa
242 IF AA43=Ø;AA43=D;GOTOa
243 IF AA44=Ø;AA44=D;GOTOa
244 IF AA45=Ø;AA45=D;GOTOa
245 IF AA46=Ø;AA46=D;GOTOa
246 IF AA47=Ø;AA47=D;GOTOa
247 IF AA48=Ø;AA48=D;GOTOa
250 GOTOb
253aNEXT D
257 NEXT C
```



```
285 GOS.z;IN."1.UW NAAM"$M
290 GOS.z;IN."2.UW NAAM"$N;P.$M;GOS.w
330 P=1;Q=Ø;GOS.y;GOS.z
340dIN."EERSTE KAART"X;GOS.z;IN."TWEEDE KAART"Y
345 IF AAX<1 OR AAY<1;GOTOc
350 IF X<>Y;IF X>Ø;IF X<49;IF Y>Ø;IF Y<49;J=#80A1;GOTOe
360cP.$30,"SVP HERHALEN    ";GOS.x;GOS.z;GOTOd
370eFOR H=1 TO 48
380 IF X=H OR Y=H;?J=I;?(J+1)=AAH
385 IF H=8 OR H=16 OR H=24 OR H=32 OR H=40;J=J+32
390 J=J+4;NEXT H
450 IF AAX=AAY;GOTOh
460 GOS.x;GOS.y;GOS.z;IF P=1;P=Ø;Q=1;P.',$N;GOS.w;G.f
470 P=1;Q=Ø;P.',$M;GOS.w
480fGOS.z;GOTOd
490hAAX=Ø;AAY=Ø;IF P=1;K=K+1;GOTOj
500 R=R+1
510jGOS.x;GOS.y;GOS.z;IF K+R=24;P."HET SPEL IS UIT";END
520 IF P=1;P.',$M;GOTOk
530 P.',$N
540kGOS.w;GOS.z;GOTOd
860yP.$30,'',$M,"....",K," PUNTEN"',$N,"....",R," PUNTEN"''" "
863 FOR S=1 TO 47
866 IF AAS<1;P."     ";GOTOv
869 IF S<10;P." ",S,"  ";GOTOv
872 P.S,"  "
875vIF S=8 OR S=16 OR S=24 OR S=32 OR S=40;P.'" "
878 NEXT S;IF AA48<1;P."  ";RETURN
879 P."48";RETURN
880wP." IS AAN DE BEURT       ";RETURN
890xFOR A=1 TO 200;WAIT;NEXT;RETURN
900zP.$30,"                                         ,",$30;RETURN
999 END
```

+=

```
500 REM WEDLOOP
610 DIM XX15,YY15;N=Ø
620 FOR A=1TO15;XXA=4;N.A;D=15
630 FOR C=4TO46 S.3;YYD=C;D=D-1;N.C
640 CLEAR Ø;P.$30;@=Ø;B=1;FOR A=ØTO43 S.3;P.B';B=B+1;N.A
650 P."     WEDLOOP";MOVE 63,3;DRAW 63,47
660 N=N+1;E=ABSRND%15+1;F=ABSRND%59+4
665 REM E IS VOOR DE NUMMERS (BIJV. VAN PAARDEN).
666 REM F IS VOOR "DRAW"
670 IF XXE>=F;GOTO 660
680 XXE =XXE+1;MOVE 4,YYE;DRAW XXE,YYE
690 IF XXE<>62;GOTO 660
700 P.$7$7$30;FOR A=1TO E;P.';N.A
705 P.$11$9$9"  ",N," PUNTEN";GOTO 660

600 REM DOOR: J.VERSTELLE
601 REM          SLEUTELBLOEMZOOM 1
602 REM          2353 RA LEIDERDORP
```

HET LAATSTE DEEL VAN DE ASSEMBLER CURSUS

Deze keer is het dan zover! De laatste aflevering van de Assembler cursus.
Wat is er te verwachten in deze aflevering:
a Het slot van de interupts
b Een doe het zelf bijlage over het puntjes tekenen in Assembler.

Allereerst de interupts deel 2

De geprogrammeerde interupt

Een geprogrammeerde interupt onderbreekt in tegenstelling tot de I/O interupt nooit een programma. Een geprogrammeerde interupt wordt dan ook pas afgehandeld als de programmeur daartoe in het programma een implicite opdracht geeft.
Een voorbeeld wat het verschil tussen een I/O interupt en een geprogrammeerde interupt. Het voorbeeld van de I/O interupt is bekend van de vorige aflevering: U leest uw krant. De postbode belt aan. Het lopende proces wordt onderbroken.
Een voorbeeld van geprogrammeerde interupt is anders: U leest uw krant. Nadat u uw krant uit heeft, gaat u kijken of de postbode al geweest is.
Dit is nu het specifieke verschil tussen een geprogrammeerde interupt en een I/O interupt.
  Een apparaat wat gek is op geprogrammeerde interupt is, is de printer. Een printer die parralel wordt aangestuurd werkt, zoals u weet, volgens het "handshake" principe. Dit handshake is in wezen niets anders dan een speciale vorm van geprogrammeerde interupt: De processor maakt de printer duidelijk dat deze geldige data (gegevens in de ASCII code) heeft. (strobe puls). De printer maakt op zijn beurt de processor duidelijk dat deze het begrepen heeft (acknowledge) , en dat de printer eventueel bezig is, met printen bij voorbeeld. (busy). De processor moet dus wachten met nieuwe data aanbieden tot de printer zich heeft terug gemeld. Dit terugmelden wordt gedetecteerd met de geprogrammeerde interupt. De 6522 VIA speelt hierbij een voorname rol Deze heeft een zogenaamd interupt register. Als er een interupt van de printer binnen komt, wordt een bitje in dit register gezet.

Door in het hoofdprogramma af te scannen of dit bitje inderdaad een 1 is, is het dus mogelijk om te kijken of dit het geval is. Het maakt dus niet uit (in principe) wanneer dit gebeurd, al is het een uur later. Het enige waar de programmeur voor moet zorgen is dat het bitje weer laag is voordat de volgende interupt om de hoek komt kijken. Dit verkaart tevens waarom de Atom er schijbaar mee stopt als er wel een VIA is aangesloten en geen printer en er wordt een (control b) gegeven met een willekeurig teken. De atom blijft wachten op de terugmelding van de printer die niet aangesloten is. En zoals u weet geven niet aangesloten apparaten in het algemeen weinig terugmeldingen.
Zoals uit het geheel wel duidelijk zal zijn geworden, wordt de IRQ noch de NMI ingangen gebruikt. Dit onderbreekt immers een lopend programma. Een geprogrammeerde interupt werkt dus alleen met een interupt vlaggen register.

De derde vorm van interupt is de DMA interupt.

Normaalgesproken schrijft allen de processor in de memory map. Toch kan het gebeuren dat dit niet snel genoeg gaat. Er is dan een mogelijkheid om de processor te isoleren van de data en adresbus. De bussen zijn nu vrij voor een ander apparaat om te gaan schrijven. Een floppy disk kan dit bijvoorbeeld doen. Onze 6502 heeft standaard geen faciliteiten voor DMA grapjes. Wil met toch gaan DMA-en, dan moet d.m.v. extra tri state buffers de hele handel worden geisoleerd. De Z80 micro processor heeft een speciale bus request ingang, die, als deze laag wordt, de hele bus vrij geeft.
Tot zover de DMA. DMA staat voor direct memory acces. (directe toegang tot het geheugen).

En dan de bijlage voor deze keer.

Deze routine tekent een puntje op X,Y waarbij X correspondeerd met ?#70 en Y met ?#71. Het werkt alleen in clear4. Probeer zelf nu eens een analyse op te zetten en de werking van dit programmatje te doorgronden. Wie weet is het bruikbaar voor u in snelle teken programma's. Het is vele malen sneller dan PLOT13,X,Y.

Wel dat was het wat de Assembler cursus betreft. Bij deze iedereen bedankt voor de positieve reacties en speciale dank voor het uit typen van de eerste afleveringen aan de heer Borghaerts.
Verder wil ik hierbij een oproep doen aan een ieder die iets over de Atom floppy drive heeft geschreven. Ik loop namelijk met plannen rond om rond de atom disk drive een soort vervolg cursus te gaan schrijven, vooral nu de acorn drives erg voordelig zijn geworden. Wie dus wat heeft, het zij programma's, hetzij documentatie, klim in de pen.

LEENDERT BIJNAGTE.

```
 10REM PUNT TEKENEN IN CLEAR4
 20REM BIJLAGE ASSEMBLER CURSUS
 30DIM LL10
 40FOR N=0 TO 10
 50  LLN=-1
 60NEXT
 70FOR N=0TO 1
 80  P=#2800;[
 90LDA #70
100STA #5A
110LDA #71
120STA #5C
130LDA #5A
140LSR A
150LSR A
160LSR A
170STA #5F
180LDA @191
190SEC
200SBC #5C
210CMP @192
220BCS LL0
230LDY @#00
240STY #60
250ASL A
260ROL #60
270ASL A
280ROL #60
290ASL A
300ROL #60
310ASL A
320ROL #60
330ASL A
340ROL #60
350ADC #5F
360STA #5F
370LDA #60
380ADC @#80
390STA #60
400LDA #5A
410AND @7
420TAY
430LDA #F7C9,Y
440LDY @#00
450ORA (#5F),Y
460STA (#5F),Y
470:LL0 RTS
480]
490 NEXT
```



```
500REM PLOT VOORBEELD
510P=#2850;[
520LDA@0
530STA#70
540STA#71
550:LL1 JSR#2800
560INC#70
570INC#71
580BNE LL1
590RTS
600]
610CLEAR4
620LINK#2850
630FOR N=0TO100
640  WAIT
650NEXT
660PRINT$12
670P=#2860;[
680:LL3LDA@00;STA#70
690:LL2 JSR#2800
700INC#70
710BNE LL2
720RTS
730:LL4
740LDA@10;STA#71;JSRLL3
750LDA@20;STA#71;JSRLL3
760LDA@30;STA#71;JSRLL3
770LDA@40;STA#71;JSRLL3
780LDA@50;STA#71;JSRLL3
790LDA@60;STA#71;JSRLL3
800RTS
810]
820CLEAR4
830LINK LL4
840END
```

LEENDERT BIJNAGTE

JOSBOX ≠ AXR 1

Toen ik in het bezit kwam van de josbox via Jan Lernout , heb ik deze direkt getest met de test die in de handleiding van de AXR 1 staat . Deze test staat ook in HOBBIT en gaat als volgt :

```
10 X=0;FOR A=≠A000 TO ≠AFFF
20 X=X:?A;NEXT A
30 PRINT X,$≠AF1C
40 END
```

Na een RUN zal de Atom na een tijdje moeten terugkeren met de melding ØSEW . Dit voor wat betreft de AXR1 . Bij mijn josbox kwam de melding 216SEW . De drie letters SEW vormen een string die in de eprom ingebakken is op locatie ≠AF1C en is vier bytes lang , inclusief het string einde karakter ≠ØD . De Ø of 216 bij de josbox is een soort checksum ; het verschil tussen AXR 1 en de josbox bleek op't eerste gezicht nogal groot en situeert zich voornamelijk binnen de instruktie RELOC . Hieronder zet ik een disassembler listing van beide eproms langs elkaar , tenminste van die stukken die verschillen :

ADRES	JOSBOX CODE	JOSBOX SYMBOOL	AXR1 ADRES	AXR1 CODE	AXR1 SYMBOOL
A8BD:	84 90	STY≠90			
A8BF:	20 FB A0	JSR≠A0FB			
A8C2:	20 88 A6	JSR≠A688			
A8C5:	20 31 C2	JSR≠C231			
A8C8:	20 8B C7	JSR≠C78B	idem als bij josbox		
A8CB:	A5 35	LDA≠35			
A8CD:	05 44	ORA≠44			
A8CF:	D0 E7	BNE≠A8B8			
A8D1:	20 AE A8	JSR≠A8AE			
A8D4:	90 E2	BCC≠A8B8			
A8D6:	20 31 C2	JSR≠C231			
A8D9:	AD FD AB	LDA≠ABFD (1)	A8D9:	20 8B C7	JSR≠C78B
A8DC:	C9 43	CMP@≠43	A8DC:	A5 36	LDA≠36
A8DE:	D0 D8	BNE≠A8B8	A8DE:	05 45	ORA≠45
A8E0:	20 8B C7	JSR≠C78B	A8E0:	D0 D6	BNE≠A8B8
A8E3:	A5 36	LDA≠36	A8E2:	20 CB C3	JSR≠C3CB
A8E5:	05 45	ORA≠45	A8E5:	20 E4 C4	JSR≠C4E4
A8E7:	D0 CF	BNE≠A8B8	A8E8:	84 91	STY≠91
A8E9:	20 CB C3	JSR≠C3CB	A8EA:	88	DEY
A8EC:	20 E4 C4	JSR≠C4E4	A8EB:	A5 90	LDA≠90
A8EF:	84 91	STY≠91	A8ED:	F0 09	BEQ≠A8F8
A8F1:	88	DEY	A8EF:	20 B7 A7	JSR≠A7B7
A8F2:	A5 90	LDA≠90	A8F2:	A8	TAY
A8F4:	F0 09	BEQ≠A8FF	A8F3:	88	DEY
A8F6:	20 B7 A7	JSR≠A7B7	A8F4:	C9 03	CMP@≠03
A8F9:	A8	TAY	A8F6:	F0 2B	BEQ≠A923
A8FA:	88	DEY	A8F8:	B1 92	LDA(≠92),Y
A8FB:	C9 03	CMP≠03	A8FA:	91 52	STA(≠52),Y
A8FD:	F0 24	BEQ≠A923	A8FC:	D1 52	CMP(≠52),Y
A8FF:	B1 92	LDA(≠92),Y	A8FE:	F0 03	BEQ≠A903
A901:	91 52	STA(≠52),Y	A900:	4C F6 CD	JMP≠CDF6
A903:	D1 52	CMP(≠52),Y	A903:	88	DEY
A905:	F0 03	BEQ≠A90A	A904:	10 F2	BPL≠A8F8
A907:	4C F6 CD	JMP≠CDF6	A906:	20 A2 A8	JSR≠A8A2

```
A90A: 88          DEY
A90B: 10 F2       BPL≠A8FF
A90D: 20 A2 A8    JSR≠A8A2
A910: 90 0E       BCC≠A920
A912: 18          CLC
A913: A5 52       LDA≠52
A915: 65 91       ADC≠91
A917: 85 52       STA≠52
A919: 90 02       BCC≠A91D
A91B: E6 53       INC≠53
A91D: C8          INY
A91E: F0 D2       BEQ≠A8F2
A920: 4C 5B C5    JMP≠C55B
A923: A5 17       LDA≠17
A925: 88          DEY
A926: F1 92       SBC(≠92),Y
A928: A5 26       LDA≠26
A92A: C8          INY
A92B: F1 92       SBC(≠92),Y
A92D: 90 D0       BCC≠A8FF
A92F: 88          DEY
A930: B1 92       LDA(≠92),Y
A932: E5 16       SBC≠16
A934: 85 54       STA≠54
A936: C8          INY
A937: B1 92       LDA(≠92),Y
A939: E5 25       SBC≠25
A93B: 90 C2       BCC≠A8FF
A93D: 85 55       STA≠55
A93F: 18          CLC
A940: A5 18       LDA≠18
A942: 65 54       ADC≠54
A944: 85 56       STA≠56
A946: A5 27       LDA≠27
A948: 65 55       ADC≠55
A94A: 91 52       STA(≠52),Y
A94C: D1 52       CMP(≠52),Y
A94E: D0 B7       BNE≠A907
A950: 88          DEY
A951: F0 AC       BEQ≠A8FF
A953: A5 56       LDA≠56
A955: B0 F3       BCS≠A94A
```

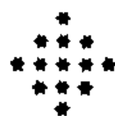
```
A909: 90 0E       BCC≠A919
A90B: 18          CLC
A90C: A5 52       LDA≠52
A90E: 65 91       ADC≠91
A910: 85 52       STA≠52
A912: 90 02       BCC≠A916
A914: E6 53       INC≠53
A916: C8          INY
A917: F0 D2       BEQ≠A8EB
A919: 20 FB A0    JSR≠A0FB ②
A91C: 4C 5B C5    JMP≠C55B
A91F: 00          BRK )
A920: 00          BRK }
A921: 80          ??? }  ③
A922: 0C          ??? )
A923: A5 17       LDA≠17
A925: 88          DEY
A926: F1 92       SBC(≠92),Y
A928: A5 26       LDA≠26
A92A: C8          INY
A92B: F1 92       SBC(≠92),Y
A92D: 90 C9       BCC≠A8F8
A92F: 88          DEY
A930: B1 92       LDA(≠92),Y
A932: E5 16       SBC≠16
A934: 85 54       STA≠54
A936: C8          INY
A937: B1 92       LDA(≠92),Y
A939: E5 25       SBC≠25
A93B: 90 BB       BCC≠A8F8
A93D: 85 55       STA≠55
A93F: 18          CLC
A940: A5 18       LDA≠18
A942: 65 54       ADC≠54
A944: 85 56       STA≠56
A946: A5 27       LDA≠27
A948: 65 55       ADC≠55
A94A: 91 52       STA(≠52),Y
A94C: D1 52       CMP(≠52),Y
A94E: D0 B0       BNE≠A900
A950: 88          DEY
A951: F0 A5       BEQ≠A8F8
A953: A5 56       LDA≠56
A955: B0 F3       BCS≠A94A
```

Verder zijn er nog drie plaatsen waar de AXR1 springt naar A919 en de JOSBOX naar C55B;

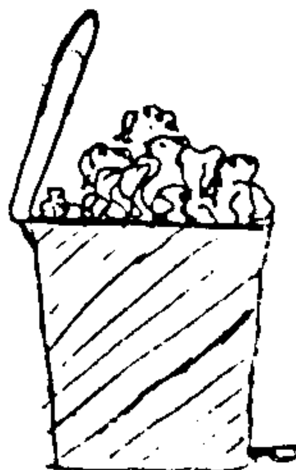
	ADRES	JOSBOX	AXR1
adres 1	A6FE	#5B	#19
	A6FF	#C5	#A9
adres 2	A8A0	#5B	#19
	A8A1	#C5	#A9
adres 3	AADA	#5B	#19
	AADB	#C5	#A9

Wat leren we nu uit deze listing ? Wel , het verschil in de routine voor de RELOC functie (staat van A8BD tot en met A956 voor zover ik weet) tussen JOSBOX en AXR1 is miniem . De instructies aangeduidt met (1) staan wel in de JOSBOX maar niet in de AXR1 . Het nut van deze instructies is mij in't geheel niet duidelijk . Immers als we gaan kijken op geheugenplaats ≠ABFD dan staat daar ≠43 , en dat staat daar voor altijd want het bevindt zich in eprom . De twee volgende regels hebben dus absoluut geen zin ; er zal toch nooit naar ≠A8B8 gesprongen worden !! Ter kompensatie van deze 7 extra bytes in de JOSBOX staan in de AXR1 ten eerst de 4 bytes gemerkt (3) .Aangezien er vóór deze blok steeds weggesprongen wordt naar ≠C55B, worden deze "instructies" nooit bereikt .Volgens mij dus volstrekt nutteloos , maar ja ik zou mij natuurlijk kunnen vergissen . Blijft nog over de extra instruktie genummerd (2) in de AXR1 . Hier wordt eerst een subroutine op ≠A0FB door lopen . Deze routine zet het X-register alsook geheugenplaats 4 dat voor de tijdelijke opslag van het X-register dient , op Ø . Dit zou van belang kunnen zijn voor de afhandeling van volgende instructies . Zeker ben ik daar niet van , dat zou nog eens nader onderzocht moeten worden . Feit is dat mijn ATOM soms kuren vertoond als de JOSBOX erin zit ; ondermeer "blijven hangen" na een foutmelding etc...

Ook de afwijkingen die onderaan op de vorige blz. staan hebben betrekking op het Ø maken van X . Immers op deze 3 plaatsen (6 bytes = 3 adressen) wordt in de AXR1 eerst naar ≠A919 gesprongen vooraleer naar ≠C55B te gaan . Het Ø maken van X zal dus wel zijn belang hebben ! Het zou natuurlijk kunnen dat er nog andere versies circuleren , dus probeer de test eens .



CUYPERS FRANK



Battery Backup

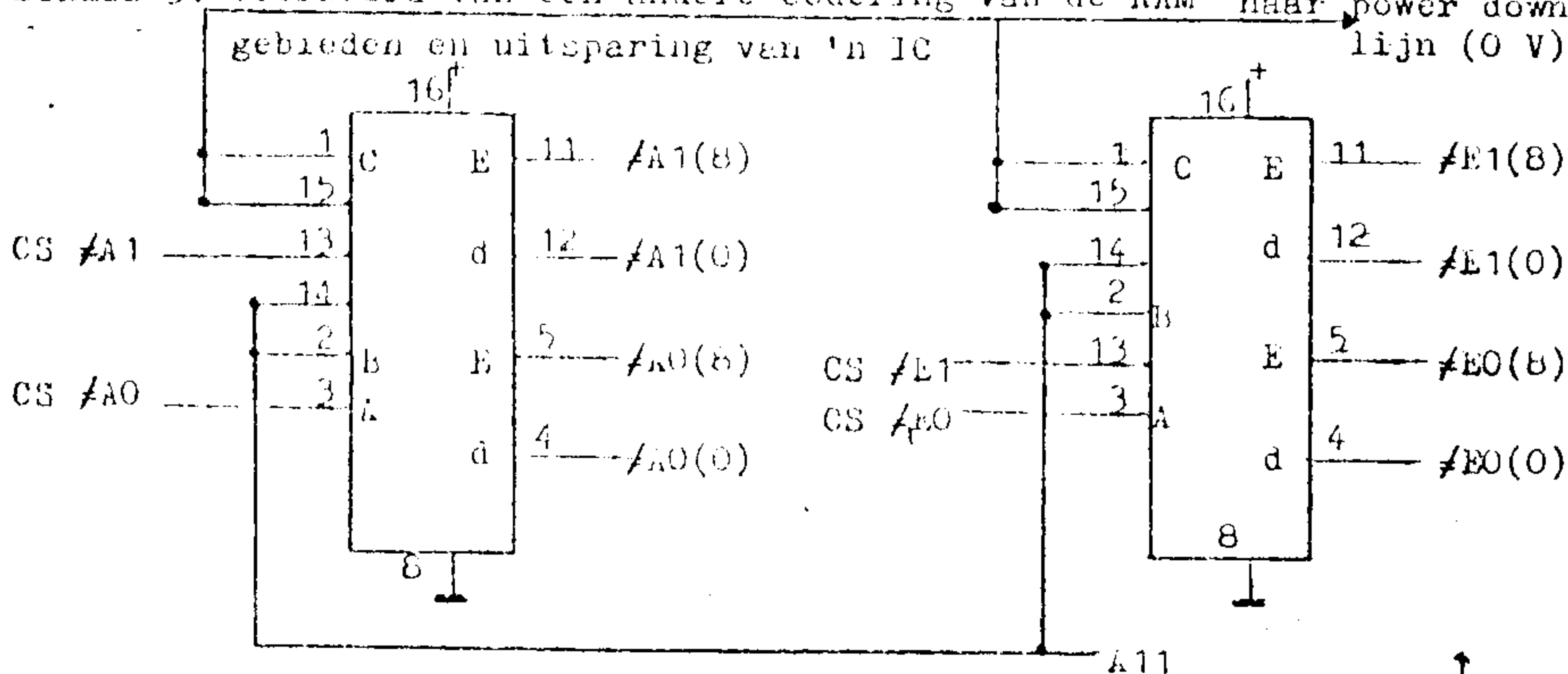
UITGEBREIDE BESCHRIJVING INCL. WIJZIGING VAN DE BATTERY BACKUP OP DE SCHAKELKAART.

Battery Backup op de schakelkaart is aangebracht om de reden dat indien de voedingsspanning afgeschakeld wordt of uitvalt, de aangesloten geheugen IC's hun inhoud niet verliezen. Dit betekend in de praktijk dat de voeding van deze IC's overgenomen wordt door een batterij en tevens worden de CS lijnen hoog gehouden. Op deze manier is men er van verzekerd dat ze kun inhoud behouden.
Bezien we het schema. De +5 V voeding brengt via D3 op punt +B een potentiaal van 4,4 V. De batterij is via D1 op het zelfde punt aangesloten, maar aangezien +4,4 groter is dan +3 spert deze diode D1. +B is de voeding voor de RAMs incl. hold IC's 3,4,5. Weerstand R1 staat parallel aan den basis-emitter diode van de transistor en hiermede is de spanning en de stroom door de weerstand bekend. De spanning over B-E diode kan max. 0,7 V bedragen en deze spanning staat ook over R1, de stroom door R1 is dus 0,6/56 Ohm = 12,5 mA. Deze 12,5 mA komen van de +5 V voeding en doorlopen zowel LED D2 als ook R2. Over R2 ontstaat een spanning van 12,5 mA x 220 Ohm = 2,75 V. Rest aldus voor de LED 5-3,45 = 1,55 volt.
De collector van T1 ligt, omdat de basis opengestuurd is, op 0 volt. Deze 0 volt wordt aan de ingangen van IC's 3,4,5 doorgegeven en staat voor de toestand; voeding aanwezig.
Daar de spanning op de collector 0 V bedraagt en de spanning op +B 4,4 V, staat over R3 een spanning van 4,4 V, welke een stroom door deze weerstand veroorzaakt van 4,4/330K = 13,3 uA, waarvan circa 3 uA benodigt is voor de ingangen van IC's 3,4,5.
Op punt +B zijn tevens de elco's C9 t/m C18 aangesloten, functioneel als buffers van de voedingsspanning en ontkoppeling van dat punt.
Indien nu de voedingsspanning van +5 V uitvalt gebeurt er het volgende: De transistor gaat sperren daar er geen sturing meer is via LED D2 en weerstand R2. De spanning op het lunt +B zal langzaam (afhankelijk van de aangesloten belasting) dalen van 4,4 V naar 2,8 V, dit komt door de restlading van bovengenoemde elco's.
Bij het afschakelen van de voeding spert D3 onmiddellijk daar op het punt +B nog het 'oude' potentiaal heerst. Diode D1 komt in geleiding zodra de spanning op punt +B beneden de 2,8 V gaat, op dit moment neemt de batterij de zaak over.
Op punt +B staat dan 2,8 volt. We hadden al gezien dat de ingangen van de IC's 3,4,5 samen circa 2,2 uA nodig hadden, zodat over R3 een spanning valt van 2,2 x 330K = 0,763 volt, zodat de restspanning op de power down lijn 2,8-0,76 = 2 volt bedraagt in standby mode.
De spanning op de ingangen van de IC's 3,4,5 bedraagt dus 2 volt (bij een batterij spanning van 3 V), dit is een erg kritische spanning daar in de MOS techniek voor een zuiver gedefinieerde '1' een spanning is vereist van >= 2 volt.
Indien de batterij dus iets aan potentiaal verliest, heeft dat tot gevolg dat deze benodigde logische '1' niet meer zuiver gedefinieerd is. We zitten dan in het verboden gebied, zoals dat heet.
Daarom kunnen we weerstand R3 van 330K beter vervangen door een 33K exemplaar. Hierdoor blijft op de PD lijn in standby mode ruim 2,7 V staan en in bedrijfs toestand (met de voeding in) loopt door T1 een stroom van circa 130 uA. Dit is natuurlijk wel een factor 100 groter dan voorheen maar dit heeft geen invloed op de werking van de schakeling.
Indien de PD lijn door een of andere sluiting aan +5 V of aan +B zou komen te liggen, kan in de huidige situatie een kortsluitstroom over

de koperbanen van PD lopen, via de transistor T1, van 1,4 Ampere. Dit kan funest zijn voor de printsporen en om dit te voorkomen kan men beter in de basis van T1 een weerstand opnemen. Deze zal indien de basisstroom zou willen toenemen, een zodanige spanningsval veroorzaken dat de IC stroom op een voor de printbanen veilige waarde begrenst wordt. Deze weerstand heeft in de bedrijfstoestand geen invloed, daar de basis stroom circa. 1 uA bedraagt. Indien we dus als basisweerstand 560 Ohm nemen, heeft dit een spanningsval tot gevolg van circa 0,5 mV, verwaarloosbaar dus.
Bovenstaande wijzigingen zijn standaard al uitgevoerd op de geheugen kaart, omdat hier het 2 V drempel probleem nog meer opspelde, daar er hier geen 4 ingangen aangestuurd moesten worden (schakelkaart) maar 8.
Bovenstaand verhaal is een beetje uitgelopen inlengte, maar ik hoop dat zeker de beginnend electronicus er iets aan heeft.

Leon Heesakkers.

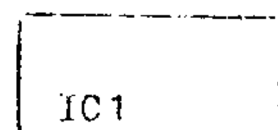
SCHETS 3: Voorbeeld van een andere codering van de RAM naar power down lijn (0 V) gebieden en uitsparing van 'n IC

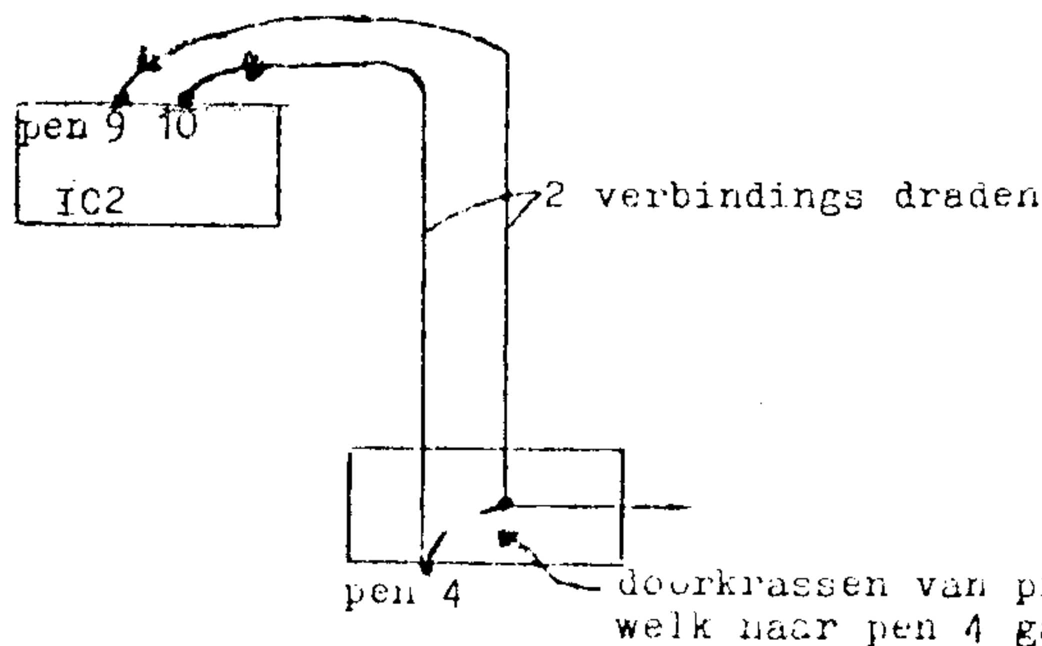


waarheidstabel

C	B	A	d	E
L	L	L	L	H
L	H	L	H	L
L	L	H	H	H
L	H	H	H	H
H	x	x	H	H

SCHETS 2: Wijziging van het soft schakeladres ≠EFFF naar ≠EFFE





De printsporen lopen aan de onderzijde.

DBASE V2.0

Inleiding.

DBASE is een programma voor de ATOM om bestanden te verwerken. Het is eigenlijk een elektronische kaartenbak. We kunnen DBASE gebruiken om ledenlijsten bij te houden, een platen- en/of boekenverzameling op te slaan, klanten met hun orders te registreren etc.
Het bijbehorende programma DPRINT V0.1 dient om gegevens in een bruikbare vorm uit te printen. Met dit programma hebben we de mogelijkheid om bv. adreslabels of voorraadlijsten te printen.

De programma's staan op de band (300 Baud) onder de namen "DBASE-V2.0"(en "DPRINT-V0.1".- VOLGENDE KEER)
We laden en starten Dbase met het commando:
*RUN"DBASE-V2.0"
dit commando laadt het programma op #8200 en linkt naar het startadres.

Beschrijving van de commando's van DBASE-V2.0.

Als we het programma starten met *RUN... of LINK#8DD9 verschijnt het keuze menu op het scherm (Dit is telkens na afloop van een commando zichtbaar (fig.1)).
De enige twee commando's die bij het begin gebruikt kunnen worden zijn CREATE en LOAD. Alle andere geven de melding:"NO FILE IN SYSTEM.".

1)CREATE.

Na het geven van het commando "C" ziet het scherm er uit als in figuur 2a. We kunnen nu beginnen met de definitie van de vorm van onze elektronische kaartenbak.
Allereerst zullen we een titelveld definieren (a) door simpelweg de door ons gewenste titel in te tikken. We willen bv een adressen bestand creeeren met als velden:

NAAM:
STRAAT:
PLAATS:
TELEFOON:

We toetsen hiertoe de veldnamen in (gevolgd door een <CR>), de overige 6 velden (we hebben een maximum van 10) laten we leeg door hier alleen een <CR> in te toetsen. Na 10 velden verschijnt automatisch het menu weer op het scherm.
NB1:Deze velddefinities kunnen niet meer ge-edit worden. Zorg er dus voor dat ze aan uw eisen voldoen.

voordat U met het invoeren van data begint. Wijzigen kan nu nog door simpelweg opnieuw het commando "C" te geven en alle velden opnieuw te definieren.
NB2:Het commando CREATE wist alle in het geheugen aanwezige data; pas hier dus mee op. Als U per ongeluk op "C" gedrukt heeft, kunt U dit weer ongedaan maken door als EERSTE en ENIGE karakter "^" in te toetsen gevolgd door een <CR> in het titelveld, waarna het menu weer verschijnt en de oude file ongewijzigd is. (Zie ook de figuren 2b en 2c).

LOAD.

Dit commando dient om een eerder gemaakt bestand van de band (of diskette) terug te lezen in de ATOM. De snelheid waarmee dit gebeurt is gelijk aan de snelheid van de COS voordat DBASE gestart werd. Als U dus op 1200 Baud wilt werken kunt u het beste DBASE opnieuw naar cassette schrijven, maar nu op 1200 Baud, met *RUN "DBASE-V2.0" wordt dan ook de rest van de lees/schrijf operaties op 1200 Baud uitgevoerd. Een andere mogelijkheid is om voordat U informatie heeft ingetikt DBASE met het Q-commando te verlaten en de COS op 1200 Baud te zetten, waarna DBASE weer gestart kan worden met LINK#8D09 . NB:Doe dit wel voordat U iets heeft ingetoetst want Quit heeft tot gevolg dat de in het geheugen aanwezige informatie niet meer door het programma als geldig wordt gezien.
Na intoetsen van "<" zien we het scherm als in figuur 3 weergegeven. Na de prompt > toetsen we nu de gewenste filenaam in en na <CR> volgt dan de PLAY TAPE boodschap, (of de file wordt geladen bij disk systemen) waarna U weer <CR> of <SPACE> toetst zoals gebruikelijk. Als het laden succesvol verlopen is geeft het systeem de melding READY , waarna de band stop gezet kan worden. Bij een fout verschijnt fig.4, we moeten dan opnieuw de laadprocedure doorlopen.

3) INSERT.

Dit commando dient om nieuwe informatie aan ons bestand toe te voegen. In het nummerveld verschijnt het nummer van het nieuwe record (de nieuwe kaart) en de veldnamen worden gelist, waarna we onze informatie kunnen intoetsen.
Het commando kan op twee manieren beeindigd worden:
a)Door alle 10 de velden in te vullen.
b)Door als laatste karakter voor de <CR> een "^" in te toetsen.
Er verschijnt in beide gevallen de boodschap:
:PRESS "^" TO GET BACK TO MENU.
Als we nog meer willen toevoegen drukken we op een willekeurige toets behalve "^" en het volgende record kan ingevoerd worden. In het andere geval zal het menu weer verschijnen.

Mogelijke fouten:
Ingevoerde gegevens moeten op het scherm passen anders worden ze niet geaccepteerd.
Verder mag de totale lengte inclusief <CR>'s niet groter zijn dan 255 karakters. Is dit toch het geval dan zal de laatst ingevoerde

regel niet opgeslagen worden en volgt er een foutmelding.

4)LIST.

De ingevoerde informatie kan bekeken worden met het LIST commando. Na toetsen van "L" zien we fig 6. Het syteem vraagt nu welk gedeelte van het bestand gelist moet worden. We hebben de volgende mogelijkheden:

a)Alles: toets <CR>

b)Van begin tot record nummer a
  toets: FROM:0<CR>
         TO   :a<CR>
  (waarbij a een nummer is tussen 0 en 253)

c)van record nummer a tot en met record nummer b
  toets: FROM:a<CR>
         TO   :b<CR>

d)van record nummer a tot en met het laatste record.
  toets: FROM:a<CR>
         TO   :253<CR>

Als het listen klaar is kunnen we weer met "^" terug naar het menu. Ook tussentijdse terugkeer naar het menu is mogelijk d.m.v. "^".

5)EDIT.

Dit commando gebruiken we als we informatie aan een al bestaand record willen toe voegen, dan wel om informatie te verbeteren. Na het geven van het commando "E" vraagt het programma om het nummer van het te wijzigen record. (zie fig.7) We toetsen het nummer nu in, gevolgd door een <CR> (0<=nummer<=253). Het bij een record behorend nummer kunnen we vaststellen m.b.v. de LIST of FIND commamdo's.
Het gewenste record wordt nu op het scherm gelist en we kunnen op de normale manier (m.b.v. copy-, delete- en cursor-toetsen) regel voor regel aanpassen. Beeindigen van het editen en terugkeer naar het menu gaat op dezelfde manier als bij het Insert commando.

6)REPLACE.

Replace gebruiken we als we een bepaald record niet meer nodig hebben en het willen vervangen door een ander record. (NB: Dit resultaat kunnen we ook bereiken d.m.v. een DELETE-commando gevolgd door een INSERT-commando).
Het systeem vraagt net als bij EDIT om het nummer van het te vervangen record. Daarna wordt het record, met het ingevoerde nummer, verwijderd en kunnen we de nieuwe informatie invoeren. Afsluiten van het commando weer als bij INSERT.

7)DELETE.

Met delete kunnen we records, die we niet meer nodig hebben,

verwijderen. Het invoeren van de gewenste nummers gaat hetzelfde als bij het list commando. Als we meer dan 5 records in een keer willen verwijderen vraagt het systeem of we dat zeker weten (fig.8).
Terugkeren naar het menu gaat weer met het commando "^".
NB:Records die met dit commando verwijderd zijn kunnen niet meer hersteld worden (zoals bijv. met OLD in BASIC). Kijk dus van te voren goed uit welke records je weggooit.

8)SORT.

We kunnen nu het door ons ingevoerde bestand sorteren en wel naar de verschillende velden.
In ons voorbeeld van een adressen bestand kunnen we dus op naam sorteren of op straat, postcode of telefoonnummer. Er is maar EEN voorwaarde waar aan voldaan moet zijn en dat is dat in alle records het veld aanwezig moet zijn waarop gesorteerd wordt (er mag ook alleen een <CR> staan). Dit is bijv. niet het geval als we bij INSERT na de definitie van PLAATS met ^ de INSERT-mode verlaten, waardoor dan het veld TELEFOON niet gedefinieerd is.
Als we wel op telefoon nummer willen sorteren moeten we dus ook een telefoonnummer invullen hetzij een leeg veld invullen door alleen op <CR> te drukken na TELFOON:.
Als we toch proberen een bestand te sorteren op een niet bestaand veld krijgen we weer de foutmelding uit figuur 4.
Na het geven van het commando "S" vraagt het systeem om het veldnummer waarop het bestand gesorteerd moet worden (fig.9). De velden zijn genummerd 0 t/m 9 (dus "NAAM:" is veldnummer 0 in ons voorbeeld en "TELEFOON:" veldnummer 3). Na het invoeren van het gewenste veld verschijnt de melding:"PLEASE WAIT WHILE SORTING". Dit wachten kan bij grotere bestanden wel enige tientallen seconden bedragen.
Als het bestand gesorteerd is keren we automatisch weer terug naar het hoofdmenu.

9)FIND.

Dit commando stelt ons in staat om records met een bepaalde inhoud te vinden. Het systeem kan zoeken tot op drie zoekstrings dus bijv: Zoek een record van PIETERSEN uit de FAZANTSTRAAT te BLARICUM. We toetsen nu bij de vragen van het syteem het volgende in:

FIRST SEARCH STRING:
PIETERSEN<CR>

SECOND SEARCH STRING:
FAZANTSTRAAT<CR>

THIRD SEARCH STRING:
BLARICUM<CR>

Waarna het systeem alle records list met die inhoud. We kunnen

echter ook niet bedoelde records vinden zoals van de heer OPBLARICUMHOOFD uit de NOGMEERFAZANTSTRAAT te PIETERSENDORP. We moeten dus een doordachte keuze maken bij het definieren van onze zoekstring.
Willen we maar een of twee zoekstrings definieren dan gaat dat als volgt:

```
FIRST SEARCH STRING:
PIETERSEN^<CR>
--------------
```

waarna alle voorkomende PIETERSEN's gelist zullen worden. Het listen kan net als bij de normale list-functie afgebroken worden met "^". Elke andere toets zal het listen gewoon voortzetten. Als de door ons opgegeven zoekstring of combinatie van zoekstrings niet in ons bestand voorkomt zal het menu weer verschijnen.

10)SAVE.

Save dient, zoals iedereen wel zal weten om het gecreeerde bestand naar tape of disk te schrijven. Het systeem vraagt weer om een filenaam en zal daarna de file saven. Na afloop volgt de boodschap "READY". D.m.v. een druk op een willekeurige toets komen we weer in het hoofdmenu.

11)*.

Dit commando geeft ons de mogelijkheid om *-commando's te geven vanuit het DBASE-programma bijv.: *. of *NOMON etc. (ook eigen gemaakte *-com's zijn natuurlijk mogelijk). Na het invoeren van een *-commando kunnen we d.m.v. "^" weer terug naar het menu of door een andere toets in te drukken nog een *-com. geven.

12)QUIT.

Hiermee beeindigen we DBASE en keren terug naar het operating systeem. Een eventueel in het geheugen aanwezige file wordt na opnieuw opstarten van DBASE niet meer als zodanig herkend !!!. Eerst SAVEN en dan pas QUIT dus. Het systeem geeft wel een waarschuwing dat dit kan gebeuren (fig.10) zodat we onze fout nog tijdig kunnen herstellen.

PØ: HET COMMANDO PRINT ZAL VOLGENDE KEER BESPROKEN WORDEN.

WILL VERHOEVEN

#DATABASE V2.0 MODE:

MENU.

1..CREATE (C)
2..INSERT (I)
3..FIND (F)
4..DELETE (D)
5..SORT (S)
6..LIST (L)
7..LOAD (<)
8..SAVE (>)
9..QUIT (Q)
10..EDIT (E)
11..REPLACE (R)
12..*-COM (*)
13..PRINT (P)

:MAKE YOUR CHOISE

FIG: 1

#DATABASE V2.0 MODE:C

CURSOR

TITELVELD

FIG: 2a

#DATABASE V2.0 MODE:C

FIG: 2b

#DATABASE V2.0 MODE:C

ADRESSEN BESTAND

NAAM :
STRAAT:
PLAATS:
TELEFOON:

:PLEASE ENTER MAINFRAME

FIG: 2c

#DATABASE V2.0 MODE: <

PLEASE ENTER FILENAME:

>

FIG: 3

#DATABASE V2.0 MODE:

ERROR - PRESS A KEY

FIG: 4

#DATABASE V2.0 MODE: I

1 ADRESSEN BESTAND

NAAM :

STRAAT:

PLAATS:

TELEFOON:

: PLEASE ENTER DATA

#DATABASE V2.0 MODE:

1 ADRESSEN BESTAND

NAAM : PIETERSEN J.

STRAAT : FAZANTSTRAAT 56

PLAATS : 0000XX BLARICUM

TELEFOON : 070-888888

↑

: PLEASE ENTER DATA

#DATABASE V2.0 MODE:L

FROM :

: ENTER RECORDNUMBERS

FIG: 6

#DATABASE V2.Ø MODE:E

WHICH RECORD:

: PLEASE ENTER RECORDNUMBER

FIG: 7

#DATABASE V2.0 MODE:D

PLEASE CONFIRM MASSIVE
DELETION [Y/N]:

FIG: 8

#DATABASE V2.0 MODE:S

SORT ON WHICH SUBSTRING [0:9]:

FIG: 9

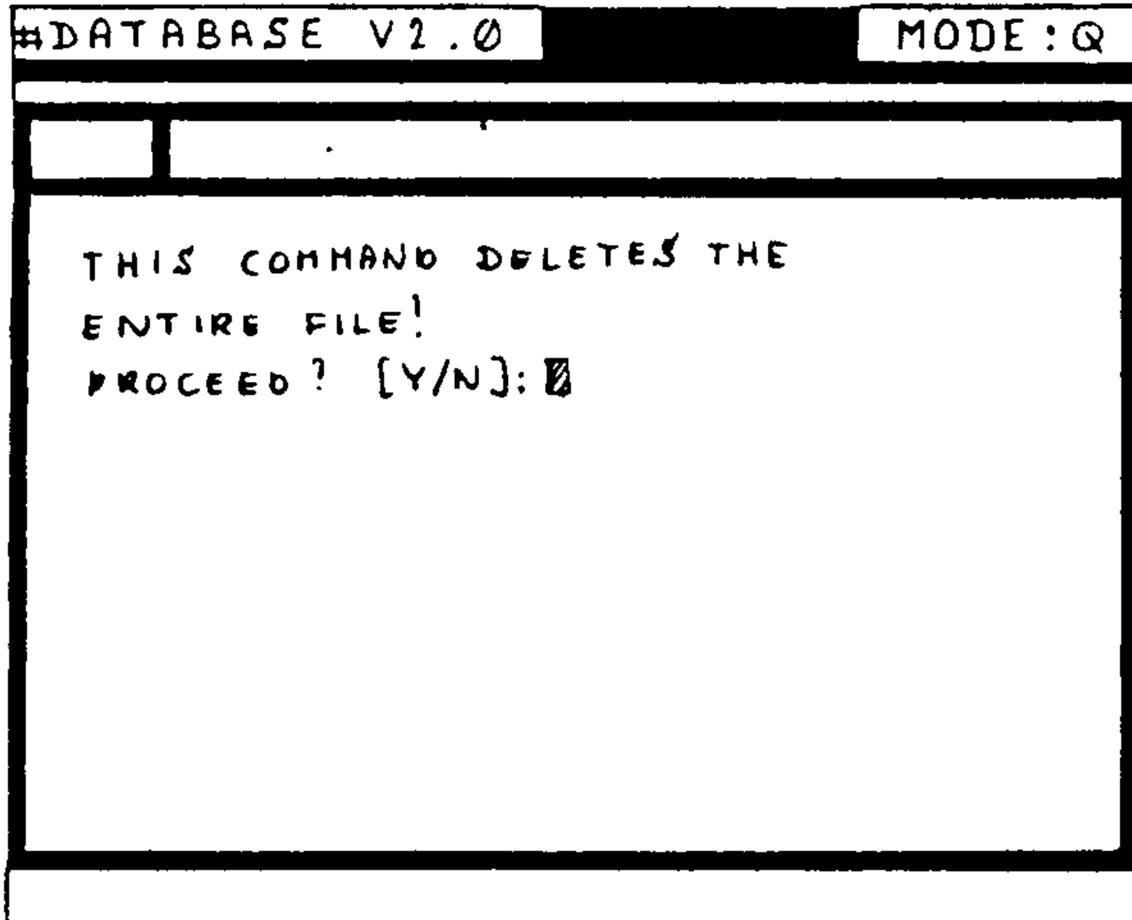


FIG:10

TOETERS & BELLEN VAN DE JOSBOX

In AcornNieuws 3 '83 werd geconstateerd dat de Josbox af en toe kuren vertoont bij gebruik van het statement READ. Dit heeft mij er toe aangezet om enkele (eigen)aardigheden van deze kit nader uit te spitten.

Om te beginnen heb ik twee slordigheidjes ontdekt in de statements COPY en XDUMP. Een opdracht als COPY ≠8201, ≠8300, ≠8200 kopieert inderdaad het gebied ≠8201 - ≠8300 naar ≠8200 - ≠82FF, maar andersom gaat het fout, dus bij COPY ≠8200, ≠82FF, ≠8201. Dan wordt ≠8200 naar ≠8201 gekopieerd, vervolgens ≠8201 (= ≠8200) naar ≠8202 enz., zodat er een soort FILL-commando ontstaat dat het gebied ≠8200 - ≠8300 vult met de waarde van ≠8200. Als het van-gebied en het naar-gebied elkaar niet overlappen gaat het echter in beide richtingen wel goed.
Het eindadres mag overigens niet ≠FFFF zijn, omdat ≠FFFF + 1 in dit geval gelijk aan nul is en dat is kleiner dan ≠FFFF, zodat COPY gewoon door gaat.
XDUMP drukt, waar mogelijk, ascii karakters af, maar met het karakter ≠7F(DEL) kun je dat beter niet doen. Probeer maar eens: XDUMP≠DDB0, ≠DDD0 en kijk wat er overblijft van de regel waar DDC0: voor staat.

Het probleem dat zich voordoet bij COPY treedt ook op bij RELOC. Bovendien werkt RELOC (uiteraard) niet op tabellen en data. Alleen pure machinetaalinstructies worden correct geRELOCeerd. Maar ook daar moet je mee oppassen. Neem b.v. RELOC≠C200, ≠CA00, ≠E200. De bytes ≠C2E1, ≠C2E6, ≠C2EB en ≠C2EF zouden ook aangepast moeten worden. Dit is principieel onmogelijk, omdat computers wel kunstmatig, maar nog lang niet intelligent zijn. Gebruik RELOC dus met overleg.

TXMOD zet de RDCH- en WRCH-vectoren terug op resp. ≠FE94 en ≠FE52. Als je, net als ik, nieuwe lees- en schrijfroutines hebt gemaakt (met auto-repeat, keyclicks, vertraagd afdrukken e.d.), dan ben je die mooi kwijt als je TXMOD uitvoert. Eigenlijk zou GRMOD de beide vectoren moeten opslaan, zodat TXMOD deze weer terug kan zetten. De Josbox gaat er kennelijk niet van uit dat je zelf ook nog uitbreidingen maakt.